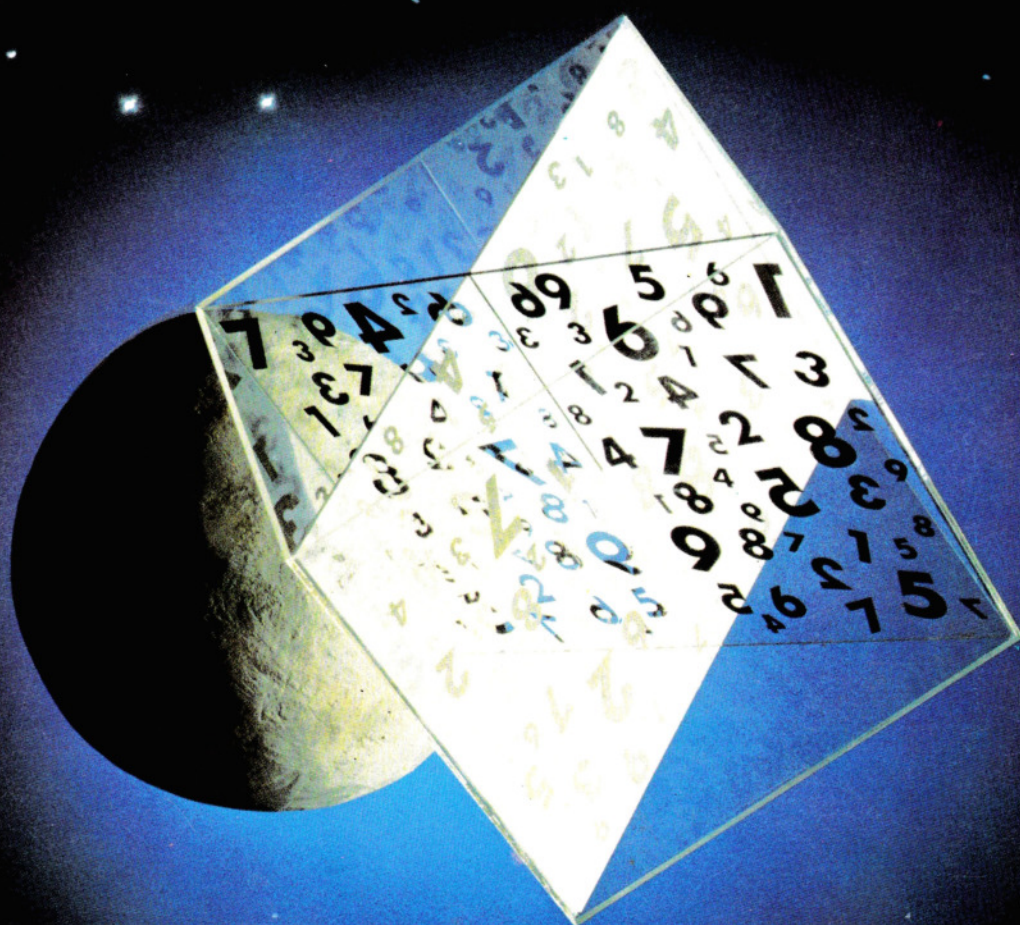
5

L. 2800

# CORSO PRATICO DI PROGRAMMAZIONE PER LAVORARE E DIVERTIRSI COL COMPUTER

ISTITUTO GEOGRAFICO DE AGOSTINI

CORSO PRATICO DI PROGRAMMAZIONE
PER LAVORARE E DIVERTIRSI COL COMPUTER

Pubblicazione a fascicoli settimanali
edita dall'Istituto Geografico De Agostini

**volume I - fascicolo 5**



*Programma di abbonamento.* Condizioni di abbonamento all'intera opera in 52 fascicoli, completa di copertine e di risguardi per la confezione dei 6 volumi dell'opera:
*a)* in un unico versamento anticipato di L. 180 000 in Italia, L. 225 000 all'estero;
*b)* in 4 versamenti trimestrali consecutivi e anticipati di L. 45 250 ciascuno.
La forma di abbonamento *b* è ammessa soltanto in Italia.
Agli abbonati all'intera opera sono riservato in dono "2 cassette di videogiochi" oppure, in alternativa, "5 cassette da registrare" (Aut. Min. conc.).
I versamenti possono essere effettuati a mezzo assegno bancario oppure sul c/c postale n. 111286 intestato all'Istituto Geografico De Agostini - Novara.
*Amministrazione, abbonamenti e servizio arretrati:* Istituto Geografico De Agostini - Novara (28100), via Giovanni da Verrazano 15 - tel. (0321) 471201-5.
Copertine e risguardi per i volumi dell'opera saranno messi in vendita a L. 6000 la copia (L. 7500 all'estero).
Le copie arretrate saranno disponibili per un anno dal completamento dell'opera e potranno essere prenotate nelle edicole o direttamente presso l'Editore. Per i fascicoli arretrati, trascorse 12 settimane dalla loro pubblicazione, è applicato un sovrapprezzo di L. 400 sul prezzo di copertina in vigore al momento dell'evasione dell'ordine. Spedizione contro rimessa di pagamento anticipato; non vengono effettuate spedizioni contrassegno.
L'Editore si riserva la facoltà di modificare il prezzo nel corso della pubblicazione, se costretto da mutate condizioni di mercato.


Registrato presso il Tribunale di Novara n. 11 in data 19-5-1984.
*Direttore responsabile:* Emilio Bucciotti

Spedizione in abbonamento postale Gruppo II/70 (Autorizzazione della Direzione provinciale delle PP.TT. di Novara).
Distribuzione A. & G. Marco - Milano, via Fortezza 27 - tel. (02) 2526.
Pubblicazione a fascicoli settimanali. Esce il martedì.
Stampato in Italia - I.G.D.A. Officine Grafiche, Novara - 208411.

*Referenze dei disegni e delle fotografie:*
Copertina: Dave King. Pagg. 129, 130 134 Peter Bentley. Pagg. 136, 138, 140, 142 Graeme Harris. Pag. 143 Archivio IGDA. Pagg. 144, 146, 148, 150 Ian Craig. Pagg. 152, 154 Dick Ward. Pagg. 156, 158, 160 Dave King.



## INPUT È STUDIATA APPOSITAMENTE PER:

Lo SPECTRUM della Sinclair (versioni 16K e 48K), il COMMODORE 64, l'ELECTRON ed il BBC della ACORN, il DRAGON 32.

*Comunque, molti dei programmi e dei testi sono adatti anche per:* lo ZX81 della SINCLAIR, il COMMODORE VIC 20 ed il TANDY COLOUR COMPUTER con 32K ed il BASIC esteso.

I seguenti simboli identificano i programmi o le spiegazioni adatte a ciascun computer:

 **SPECTRUM**

 **COMMODORE 64**

 **ELECTRON e BBC**

 **DRAGON 32**

 **ZX81** **VIC 20**

 **TANDY TRS80 COLOUR COMPUTER**

# E DOPO CHE FARE?

- USO DI **INPUT** E DEI **PROMPT**
- METODI PIÙ RAPIDI IMPIEGANDO **INKEY$** O **GET$**
- DISEGNARE SULLO SCHERMO
- PAROLA D'ORDINE SEGRETA

**Perché un computer risponda alle nostre aspettative, occorre prima comunicargli quali esse siano. Esistono più sistemi per far ciò, secondo il tipo di programma**

Salvo pochissime eccezioni, i computer non sono capaci di lavorare se non vengono prima istruiti. In tutti i programmi, siano essi di gioco, di contabilità o di applicazioni scientifiche, è necessaria una qualche immissione di dati da parte dell'operatore. Ma le informazioni che si possono fornire al computer sono di molti generi, così come molti sono i vari modi per immetterle.

L'informazione può essere semplice, come la pressione di un tasto per spostare un missile, ma anche molto complessa, come durante l'immissione di dati per i programmi di calcolo o per esperimenti scientifici. Nei giochi, è essenziale che il computer reagisca velocemente, mentre in altre applicazioni è più importante l'abilità nel trasformare il dato digitato prima di affidare quest'ultimo alla memoria del computer.

## GLI INPUT PIU' SEMPLICI

Per illustrare l'uso di INPUT, si ricorre quasi sempre a un programma molto semplice, composto da poche linee:

```
10 PRINT "COME TI CHIAMI?"
20 INPUT N$
30 PRINT "CIAO,□";N$
```

Quando il computer incontra l'istruzione INPUT, attende l'immissione di qualcosa. La sequenza di tasti digitata, in genere fino al primo RETURN o ENTER, viene depositata in una variabile (nel nostro esempio, in N$).

In questo caso la variabile è una stringa, ma il discorso vale anche per le variabili numeriche. Il seguente programma mostra come allestire un elenco di cinque nomi ed età usando semplici comandi INPUT. (Vengono usate due matrici: se non si ha familiarità con il loro funzionamento, si veda alle pagine 152-155).

```
5 DIM NOME$(5), ETÀ(5)
10 FOR N=1 TO 5
20 PRINT "NOME:"
30 INPUT NOME$(N)
40 PRINT "ETÀ:"
50 INPUT ETÀ (N)
60 NEXT
70 FOR N=1 TO 5
80 PRINT NOME$(N);"□HA□";ETÀ(N);"□
   ANNI"
90 NEXT
```

```
5 DIM NOME$(S) ETÀ(5)
10 FOR N=1 TO 5
20 PRINT "NOME:"
30 INPUT NOME$(N)
40 PRINT "ETÀ:"
50 INPUT ETÀ(N)
```

```
60 NEXT N
70 FOR N=1 TO 5
80 PRINT NOME$(N);"□HA";ETÀ(N);"□ANNI"
90 NEXT N
```

```
5 DIM n$ (5,10): DIM a (5)
10 FOR n=1 TO 5
20 PRINT "NOME:□"
30 INPUT n$ (n)
40 PRINT "Età:□"
50 INPUT a (n)
55 CLS:NEXT n
60 FOR n=1 TO 5
70 PRINT n$(n); "□ha□"; a(n); "□anni"
80 NEXT n
```

Sullo ZX81, si batta tutto in maiuscole. Al posto delle linee 5 e 55, si aggiunga:

```
5 DIM N$ (5,10)
7 DIM A (5)
55 CLS
57 NEXT N
```

È importante fornire il tipo giusto di informazione: se si tentasse di inserire un nome quando è richiesta una età, il programma si fermerebbe con un messaggio di errore (eccetto che sull'Acorn). Rieseguire il RUN può causare la perdita di tutto quel che si è immesso: non è un grave problema con solo cinque elementi, ma può diventare molto fastidioso con una maggiore quantità di dati.

## COSA SONO I PROMPT

Il programma appena visto può sembrare piuttosto semplice, tuttavia esso contiene, nelle prime due PRINT, un elemento molto importante: un *prompt*. Questo termine indica semplicemente il messaggio che invita l'operatore a immettere un'informazione.

Cancellando le linee 2Ø e 4Ø, il programma gira ancora e, nella maggior parte dei computer, sullo schermo appare ad esempio un punto interrogativo. Avendo familiarità col computer, a volte basta una simile richiesta di INPUT per ricordare che dobbiamo immettere qualcosa, ma ciò non basta a spiegare il tipo di informazione. Immaginiamo di non conoscere il programma né il suo funzionamento e, per giunta, di non avere alcuna esperienza sul computer usato: un punto interrogativo offre soltanto perplessità.

Un buon uso dei prompt è utile anche se si è alle prese con i propri programmi, dato che è facilissimo dimenticare il preciso formato col quale va immessa un'informazione. Un buon esempio di ciò si ha nell'immissione di date. La maggior parte dei programmi di contabilità, ma anche molti altri, richiedono l'immissione di date. In molti casi, questa informazione è usata dal programa in routine di ricerca o per selezionare gli elementi in ordine cronologico. Nel fare ciò, bisogna assicurarsi di immettere la data sempre col solito formato.

Un metodo antico, molto diffuso, consiste nell'usare sei cifre, a volte con barre oblique o punti, per separare giorno, mese e anno. Una tipica routine di INPUT può avere questo aspetto:

```
100 PRINT "IMMETTERE LA DATA (NEL
    FORMATO GG/MM/AA)"
110 INPUT D$
```

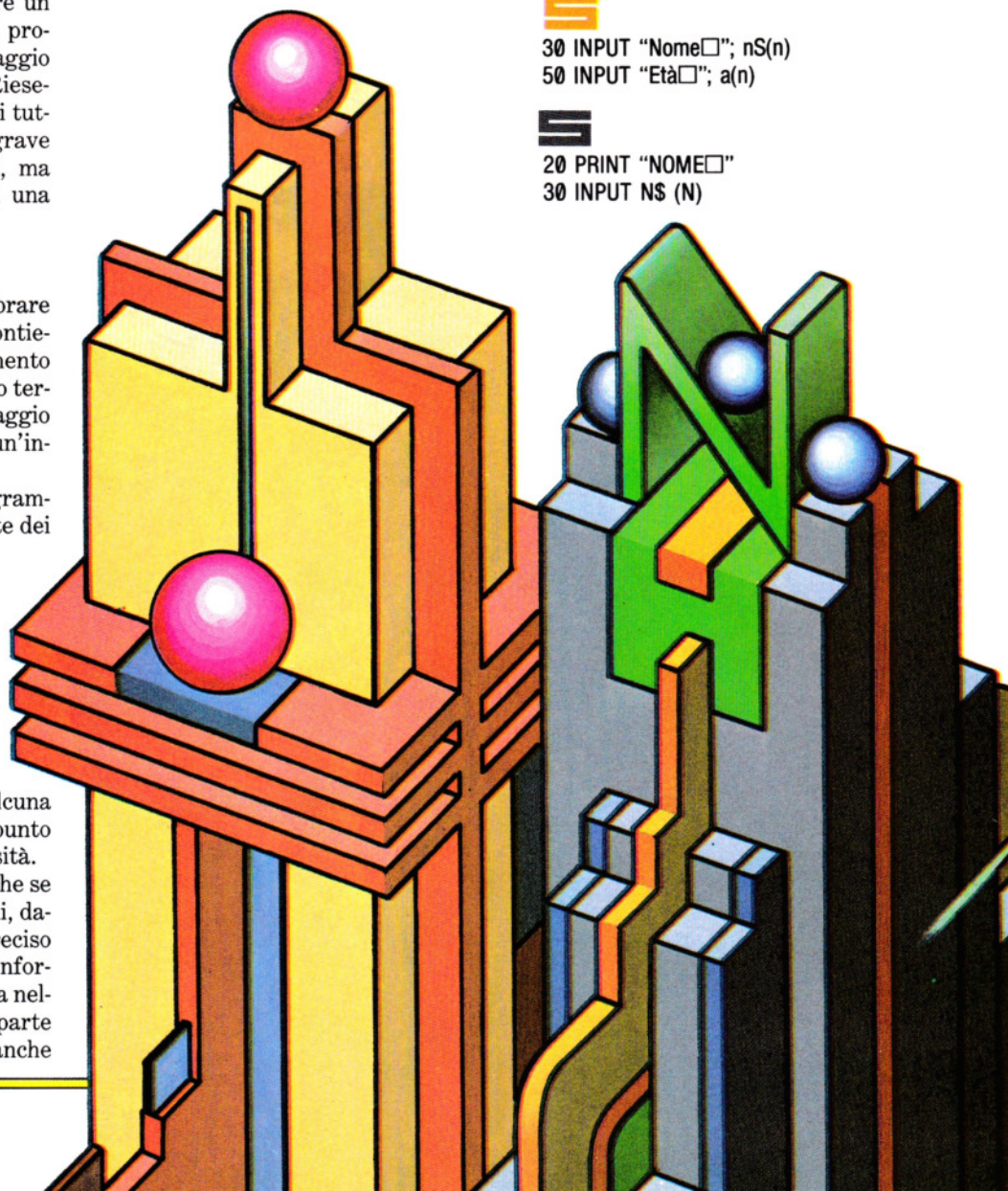
Questo ricorda non solo che occorre inserire una data, ma anche il formato col quale digitarla. Se la data è il 3 Settembre 1945, si scriverebbe allora Ø3/Ø9/45. È indifferente quale formato viene scelto, purché l'utente sia avvertito di quale esso sia. L'uso del comando INPUT nei programmi mostrati finora è piuttosto semplice. Ma può diventare anche più semplice, poiché in molti computer il prompt può essere incorporato nello stesso comando INPUT. (Sul Commodore, la lunghezza del prompt è limitata a 2Ø caratteri). Tornando al programma del nome e dell'età, si possono cancellare le linee 2Ø e 4Ø e riscrivere la 3Ø e la 5Ø così:

```
30 INPUT "NOME□" NOME$(N)
50 INPUT "ETÀ□"ETÀ(N)
```

```
30 INPUT "NOME□"; NOME$(N)
50 INPUT "ETÀ□"; ETÀ(N)
```

```
30 INPUT "NOME"; NOME$(N)
50 INPUT "ETÀ";ETÀ(N)
```

```
30 INPUT "Nome□"; nS(n)
50 INPUT "Età□"; a(n)
```

```
20 PRINT "NOME□"
30 INPUT N$ (N)
```

```
40 PRINT "ETÀ□"
50 INPUT A(N)
```

Si faccia caso agli spazi all'interno degli apici. In molti computer, il primo carattere digitato come risposta appare subito dopo il prompt: un spazio serve a tenere le cose più ordinate. Nel Dragon e nel Tandy, comunque, lo spazio viene inserito automaticamente.

Non è questo il solo modo di semplificare la routine. In genere, è consentita l'immissione anche di più elementi, usando un singolo comando INPUT: il solo computer su cui ciò non è possibile è lo ZX81. I nomi delle variabili in cui depositare i valori sono separati da virgole. Possiamo dunque sostituire le linee 30 e 50 qui sopra (cancellando la 50) con questa sola linea:

```
30 INPUT "IMMETTERE NOME ED ETÀ□"
   NOME$(N),ETÀ(N)
```

```
30 INPUT "IMMETTERE NOME ED ETÀ□";
   NOME$(N),ETÀ(N)
```

```
30 INPUT "IMMETTERE NOME ED ETÀ";
   NOME$(N),ETÀ(N)
```

```
30 INPUT "Immettere nome ed età□";
   n$(n),a(n)
```

Il metodo per separare le informazioni immesse varia da un tipo di computer all'altro: sullo Spectrum si preme ENTER alla fine di ciascun dato, mentre sul Dragon, sul Tandy, sull'Acorn e sul Commodore, si può, in alternativa, digitare nome e età, separandoli da una virgola, e solo allora premere ENTER.

Se si usa un solo prompt, è importante che esso segnali in modo chiaro il tipo ed il formato dell'informazione richiesta.

In alternativa, sull'Acorn e sullo Spectrum, si può sempre tornare a suddividere i prompt: purché ciascuno sia racchiuso tra apici, il computer non li tratta come variabili. Perciò, la linea 30 può di nuovo venir riscritta:

```
30 INPUT "NOME□" NOME$(N), "ETÀ□"
   ETÀ(N)
```

```
30 INPUT "NOME" ; n$(n); "età"; a(n)
```

Ad ogni modo, si noti che questo metodo non è applicabile al Commodore, al Dragon, al Tandy o allo ZX81.

Per ottenere uno schermo ordinato, i prompt vengono spesso posizionati mediante normali comandi TAB (vedere alle pagine 117-123).

## L'INPUT DI UN'INTERA LINEA

Il maggior problema, nell'impiego della INPUT, si incontra quando occorre immettere stringhe contenenti virgole o spazi iniziali. Volendo usare una sola variabile per un intero indirizzo, per esempio, sarebbe comodo poter includere delle virgole. Uno o più spazi iniziali possono contribuire a un maggior ordine sullo schermo.

La maggior parte dei computer, però, ignora gli spazi iniziali, benché non abbia problemi con gli spazi tra le parole. Spesso, ignora anche qualsiasi cosa segua una virgola, perciò ampie porzioni di un'informazione potrebbero andare perdute.

Fortunatamente, alcuni computer hanno un comando per risolvere questo problema. Lo Spectrum possiede un modo per aggirarlo.

La soluzione più comune è di racchiudere il dato immesso tra virgolette. Lo Spectrum, ad esempio, visualizza automaticamente le virgolette se la INPUT riguarda una variabile stringa. Sugli altri computer, invece, queste devono essere digitate dall'operatore e questo fatto dovrebbe essere evidenziato nel relativo prompt.

Un'altra soluzione, disponibile sull'Acorn, sul Dragon e sul Tandy, è di usare il comando INPUT LINE o LINE INPUT. Il suo uso è esattamente lo stesso di INPUT:

```
10 INPUT LINE "Immettere il proprio
   indirizzo□", A$
```

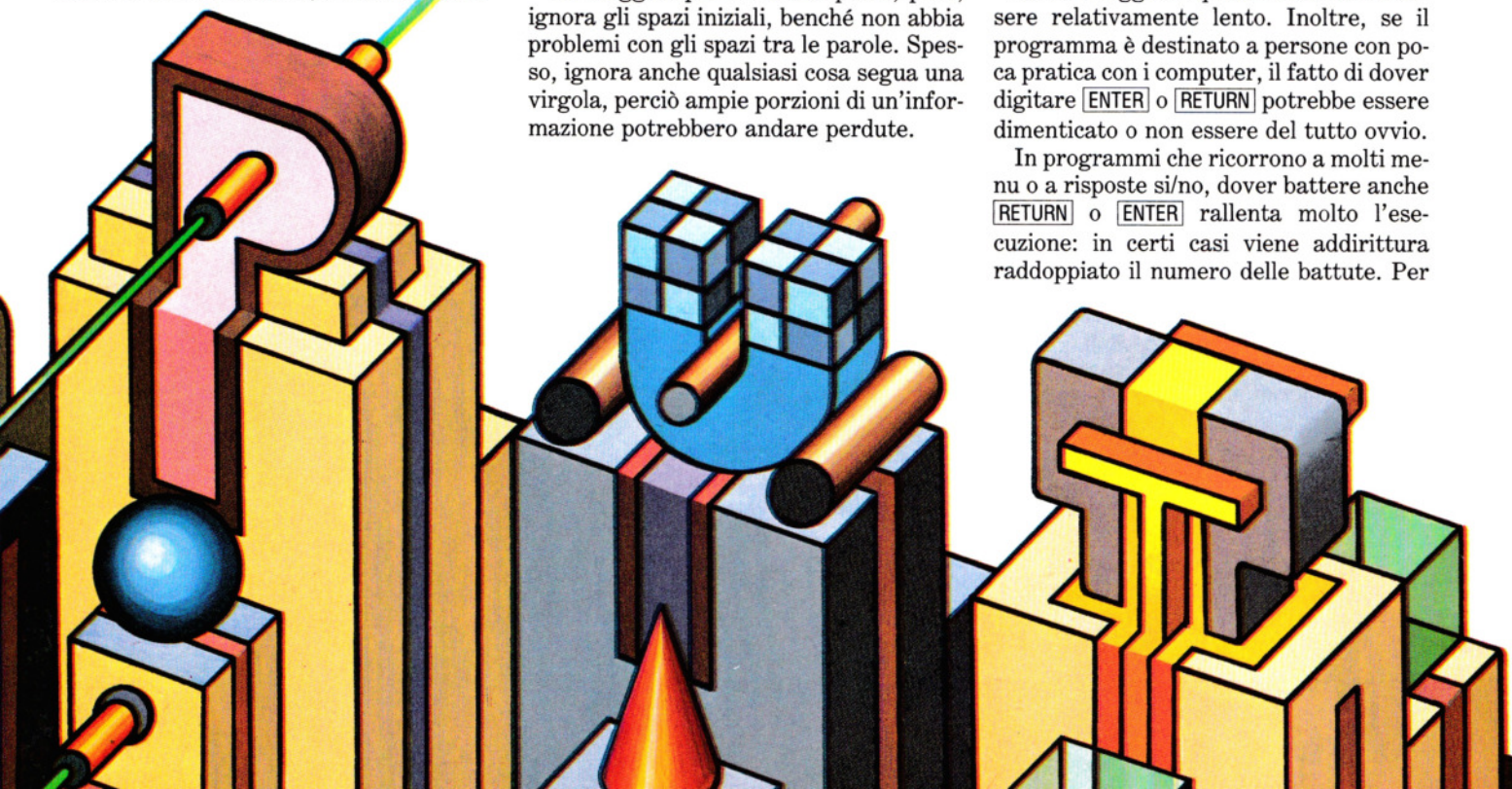
```
10 LINE INPUT "Immettere il proprio
   indirizzo□"; A$
```

Adesso, tutto quanto precede il primo RETURN, comprese virgole e spazi iniziali, viene depositato nella variabile. Non essendo necessario digitare le virgolette, diminuiscono gli errori d'immissione.

## SVELTIRE GLI INPUT

Il vantaggio principale di INPUT e INPUT LINE è la possibilità di correggere quanto si sta immettendo, fino al momento del RETURN o dell'ENTER. Perciò, se capita di sbagliare una o più battute, o se si è semplicemente cambiata idea, basta cancellare il dato e ripartire.

Lo svantaggio di questo sistema è di essere relativamente lento. Inoltre, se il programma è destinato a persone con poca pratica con i computer, il fatto di dover digitare ENTER o RETURN potrebbe essere dimenticato o non essere del tutto ovvio.

In programmi che ricorrono a molti menu o a risposte si/no, dover battere anche RETURN o ENTER rallenta molto l'esecuzione: in certi casi viene addirittura raddoppiato il numero delle battute. Per

evitare tutto ciò, c'è un metodo per far leggere al computer il tasto premuto nel momento in cui ciò avviene.

Il comando usato è INKEY$ sugli apparecchi Sinclair, Tandy e Dragon, GET$ o INKEY$ sugli Acorn e GET sui Commodore.

Quando il computer incontra una di queste istruzioni, controlla se si è premuto un tasto e poi passa il corrispondente carattere alla variabile. Questo metodo è abituale nelle risposte si/no a domande come "Vuoi un'altra partita?" (vedere a pagina 35). La sua forma può essere:

```
100 LET A$=GET$
```

```
100 GET A$ : IF A$=""THEN GOTO 100
```

```
100 LETA$=INKEY$ : IF A$=""
   THEN GOTO 100
```

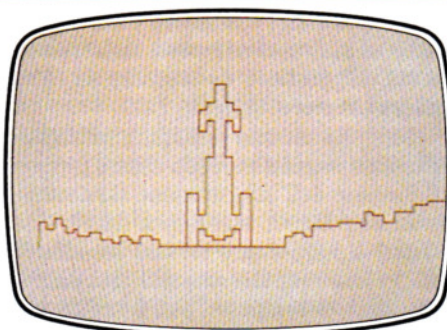
```
100 LET A$=INKEY$
110 IF A$="" THEN GOTO 100
```

Si noti che, sull'Acorn, GET$ fa attendere il computer finché non viene premuto un tasto, mentre gli altri apparecchi devono essere istruiti per l'attesa.

Si può usare qualsiasi variabile stringa: A$ è solo un esempio. Il computer si ferma alla linea 100. Premendo R, A$ conterrà la lettera R. Si può immettere un numero o uno spazio o qualsiasi carattere sulla tastiera, o anche tasti come ENTER. Anche se l'immissione può virtualmente essere qualsiasi cosa, la lunghezza deve essere di un solo carattere. Appena viene premuto un tasto, il computer procede.

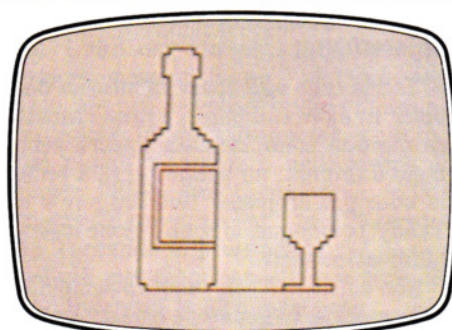
## DISEGNARE SULLO SCHERMO

Il prossimo programma disegna sullo schermo con l'uso di quattro tasti (questo programma non gira sullo ZX81). Sull'Acorn, il Dragon e lo Spectrum, si usano per disegnare Z, X, P e L. Sul Commodore, i tasti cursore. Si può anche premere 2 per disegnare nel colore dello sfondo (la linea risulta invisibile) e 1 per tornare a tracciare linee visibili.

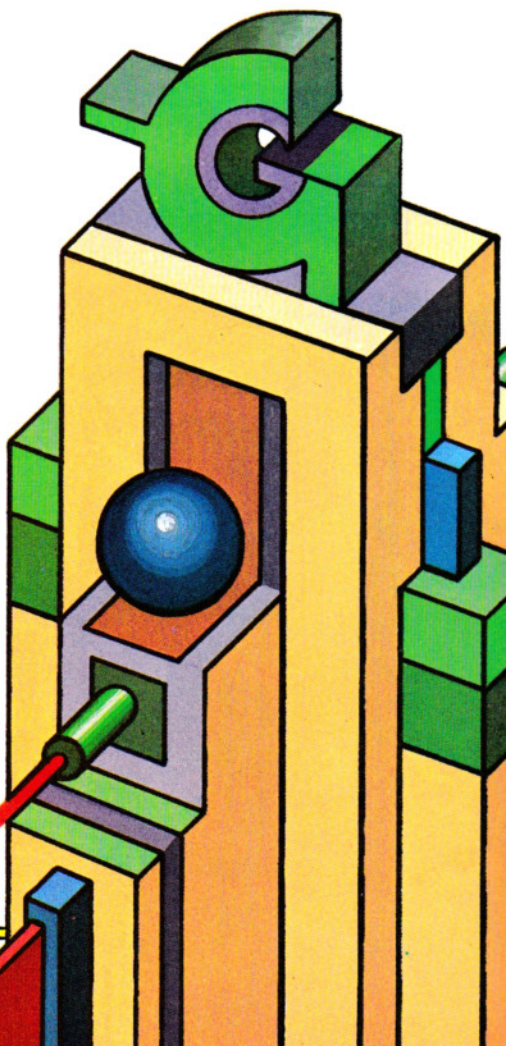
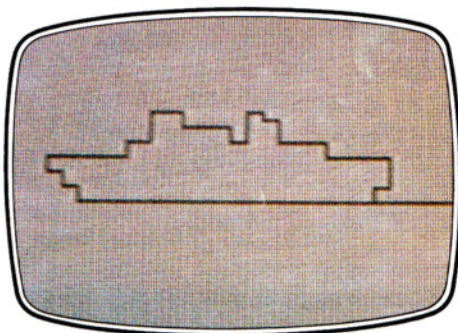
```
10 MODE 1
20 LET X=500: LET Y=500
30 MOVE X,Y
40 REPEAT
50 LET A$=GET$
60 IF A$="P" THEN LET Y=Y+4
70 IF A$="L" THEN LET Y=Y-4
80 IF A$="Z" THEN LET X=X-4
90 IF A$="X" THEN LET X=X+4
```

**Si può disegnare un'immagine sullo schermo controllandola dalla tastiera con un breve e semplice programma**

```
100 IF A$="1" THEN GCOL0,3
110 IF A$="2" THEN GCOL0,0
120 DRAW X,Y
130 UNTIL A$="□"
```

```
10 FOR Z=0 TO 69:READ X
20 POKE 832+Z,X:NEXT Z :GOTO 90
30 DATA 169,29,141,24,208,169,59,141,17,
   208
40 DATA 169,32,133,252,169,0,133,251,160,
   0,169,0,145,251
50 DATA 200,208,251,24,165,252,201,63,
   240,4,230,252,208,236
60 DATA 162,0,169,0,157,0,64,232,224,63,
   208,248
70 DATA 162,0,169,13,157,0,4,157,0,5,157,0,
   6,157,232,6,232,208,241,96
90 SYS 832
100 SC=8192:XX=100:YY=100:CO=1
110 LET Y=YY:LET X=XX
120 GET A$:IF A$="" THEN GOTO 120
130 IF A$="▮" THEN LET X=X-1
140 IF A$="▮" THEN LET X=X+1
150 IF A$="▮" THEN LET Y=Y-1
160 IF A$="▮" THEN LET Y=Y+1
165 IF A$="1" THEN CO=1
166 IF A$="2" THEN CO=2
170 LET L=SC+(INT(Y/8)*320+8*INT(X/8)
   +(Y AND 7))
180 IF L<8192 OR L>16191 THEN GOTO
   110
190 LET XX=X:LET YY=Y
200 IFCO=1THEN POKE L, PEEK(L) OR (2 ↑
   (7-(XX AND 7)))
210 GOTO 110
```

```
10 PMODE4,1:PCLS:SCREEN 1,1
20 LET X=100:LET Y=100
40 LET X1=X:LET Y1=Y
50 LET A$=INKEY$
60 IF A$="P" THEN LET Y=Y1-4
70 IF A$="L" THEN LET Y=Y1+4
80 IF A$="Z" THEN LET X=X1-4
90 IF A$="X" THEN LET X=X1+4
```

**Il programma permette di spostare la linea in diverse direzioni premendo particolari tasti**

```
100 IF A$="1" THEN COLOR 5
110 IF A$= "2" THEN COLOR 0
120 IF A$="□" THEN STOP
130 LINE (X,Y)-(X1,Y1),PSET
140 GOTO 40
```

```
10 INK 2
20 PLOT 127,87
30 IF INKEY$="p" THEN DRAW 0,2
40 IF INKEY$="l" THEN DRAW 0,-2
50 IF INKEY$="z" THEN DRAW -2,0
60 IF INKEYS="x" THEN DRAW 2,0
70 IF INKEY$="1" THEN INK 2
```

**Esercitiamoci a disegnare ricopiando queste o altre forme attenendoci alle istruzioni date nel testo**

**Si noti che le diagonali, non potendosi premere due tasti per volta, vanno disegnate come serie di piccoli scalini**

```
80 IF INKEY$="2" THEN INK 7
90 IF INKEY$="□" THEN STOP
100 PAUSE 10
110 GOTO 30
```

Una routine di questo tipo è molto utile nella grafica e nei giochi. Ogni linea viene tracciata finché i tasti sono premuti, ma si noti che il computer accetta un solo tasto per volta, perciò è molto difficile disegnare linee diagonali, dato che sono costituite da una serie di piccoli passi.

L'uso di INKEY$ o GET$, per singoli caratteri, è molto comodo nei programmi dotati di menu. Il seguente programma visualizza un tipico menu per un programma di archiviazione.

```
10 DATA CREAZIONE FILE, IMMISSIONE,
   VISIONE, MODIFICA, RICERCA, STAMPA,
   LETTURA, SCRITTURA, FINE
15 RESTORE
20 FOR N=1 TO 9
30 READ TESTATA$
40 PRINT TAB(5);N;TAB(10);TESTATA$
50 NEXT N
60 PRINT:PRINT TAB(5)"QUALE SCEGLI>"
70 LET A$=GET$
80 IF A$="1" THEN GOSUB 1000
90 IF A$="2" THEN GOSUB 2000
100 IF A$="3" THEN GOSUB 3000
110 IF A$="4" THEN GOSUB 4000
120 IF A$="5" THEN GOSUB 5000
130 IF A$="6" THEN GOSUB 6000
140 IF A$="7" THEN GOSUB 7000
150 IF A$="8" THEN GOSUB 8000
160 IF A$="9" THEN GOSUB 9000
170 GOTO 15
```

Si cancelli la linea 70 sul programma dell'Acorn, sostituendola con:

```
70 GET A$:IF A$="" THEN GOTO 70
```

Si cancelli la linea 70, sostituendola con:

```
70 LET A$=INKEY$: IF A$="" THEN GOTO
   70
```

Si cancellino le linee dalla 10 alla 70, sostituendole con:

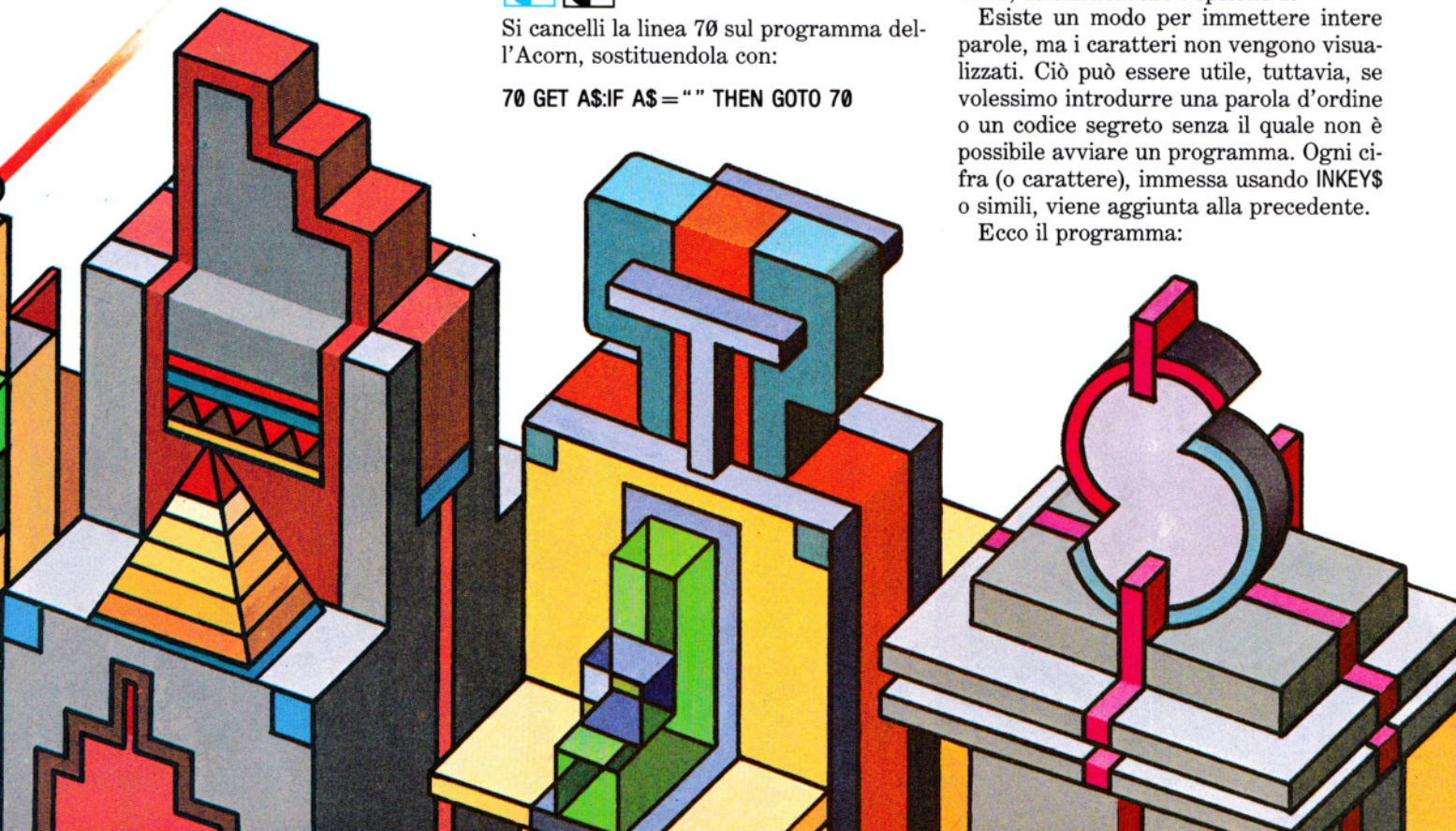
```
10 DATA "Creazione file", "Immissione
   record", "Visione record", "Modifiche di un
   record", "Ricerca di un record", "Stampa
   archivio", "Lettura file", "Scrittura file",
   "Fine"
15 RESTORE
20 FOR n=1 TO 9
30 READ h$
40 PRINT TAB 5; n; TAB 1; h$
50 NEXT n
60 PRINT: PRINT TAB 5; "Quale scegli>"
70 LET A$=INKEY$ : IF A$="" THEN
   GOTO 70
```

Con questo programma, il computer va direttamente alla subroutine corrispondente appena si preme un tasto, purché sia compreso tra 1 e 9. In caso contrario, la linea 170 visualizza nuovamente il menu, invitando ancora a scegliere.

## LA PAROLA D'ORDINE SEGRETA

Il programma precedente funziona perfettamente se si hanno meno di nove opzioni. Se però tentassimo di scrivere 10, il computer, che accetta una sola cifra per volta, selezionerebbe l'opzione 1.

Esiste un modo per immettere intere parole, ma i caratteri non vengono visualizzati. Ciò può essere utile, tuttavia, se volessimo introdurre una parola d'ordine o un codice segreto senza il quale non è possibile avviare un programma. Ogni cifra (o carattere), immessa usando INKEY$ o simili, viene aggiunta alla precedente.

Ecco il programma:

```
10 PRINT "IMMETTERE CHIAVE"
15 REPEAT
20 LET K$=GET$
30 LET P$=P$+K$
40 UNTIL LEN(P$)=7
50 IF P$<> "MARAMEO" THEN END
60 PRINT "O.K."
70 REM (resto del programma)
```

```
10 PRINT "▣▣▣▣▣▣▣▣▣"
   TAB(13) "IMMETTERE CHIAVE"
20 FOR Z=1 TO 7
30 GET K$:IF K$="" THEN GOTO 30
40 LET P$=P$+K$
50 NEXT Z
60 IF P$<> "MARAMEO" THEN END
70 PRINT "▢" TAB(13) "□CHIAVE
   CORRETTA□□□"
80 REM (RESTO DEL PROGRAMMA)
```

```
10 PRINT "IMMETTERE CHIAVE"
20 LET K$=INKEY$:IF K$="" THEN GOTO
   20
30 LET P$=P$+K$
40 IF LEN(P$)<>7 THEN GOTO 20
50 IF P$<> "MARAMEO" THEN STOP
60 PRINT "O.K."
70 REM (RESTO DEL PROGRAMMA)
```

Sullo ZX81, si scriva interamente in maiuscole e si sostituisca la linea 40 con:

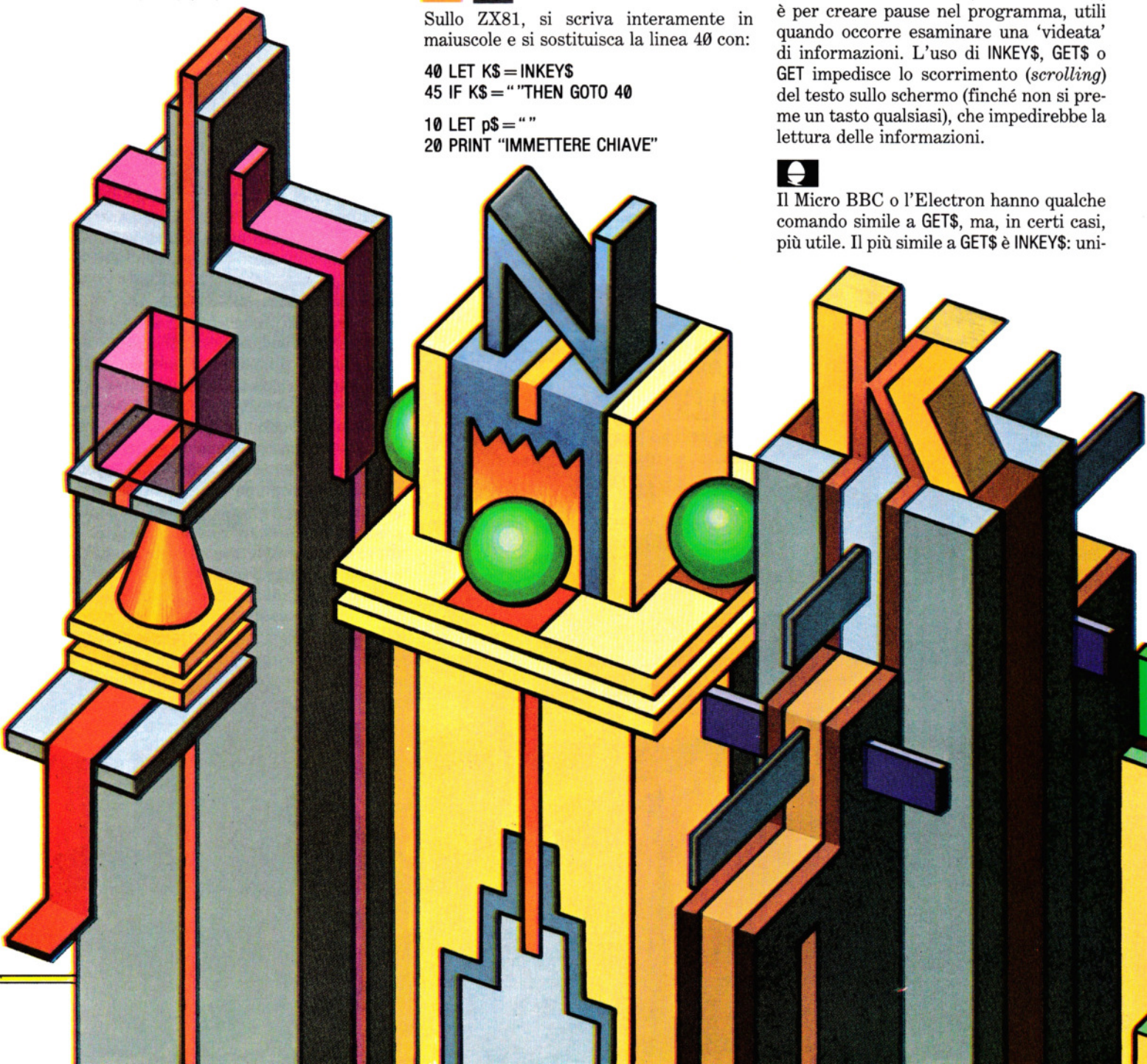
```
40 LET K$=INKEY$
45 IF K$=""THEN GOTO 40
```

```
10 LET p$=""
20 PRINT "IMMETTERE CHIAVE"
30 PAUSE 0
40 LET k$=INKEY$: IF k$="" THEN GOTO
   40
50 LET p$=p$+k$
60 IF LEN p$<>7 THEN GOTO 30
70 IF p$<>"marameo" THEN STOP
80 PRINT "O.K."
90 REM (resto del programma)
```

Questa routine parte con una stringa vuota (P$) e va aggiungendo ad essa un carattere alla volta, finché la lunghezza della parola d'ordine non sia corretta. Dato che i caratteri digitati non sono visibili sullo schermo, ciò impedisce che occhi indiscreti carpiscano la parola d'ordine.

Un altro uso di INKEY$ (e comandi simili) è per creare pause nel programma, utili quando occorre esaminare una 'videata' di informazioni. L'uso di INKEY$, GET$ o GET impedisce lo scorrimento (*scrolling*) del testo sullo schermo (finché non si preme un tasto qualsiasi), che impedirebbe la lettura delle informazioni.

Il Micro BBC o l'Electron hanno qualche comando simile a GET$, ma, in certi casi, più utile. Il più simile a GET$ è INKEY$: uni-

ca differenza è che, mentre un comando GET$ fa attendere il computer all'infinito finché non si preme un tasto, con INKEY$ si specifica un limite di tempo.

INKEY$ è sempre seguito da un numero tra parentesi, che indica l'intervallo di attesa in centesimi di secondo. Per ottenere una pausa di cinque secondi, per esempio, si scriverebbe: A$ = INKEY$(500).

Premendo un tasto prima della fine del periodo, il computer prosegue con il programma: il limite della pausa è soltanto un massimo. Se non viene premuto nessun tasto prima della scadenza, il risultato è una stringa nulla (" "). INKEY$(Ø) fa controllare al computer la tastiera, ma senza nessuna attesa: è utile nei giochi, dove ciò che più conta è la velocità.

GET è simile a GET$, ma, invece di un carattere, produce il codice ASCII del tasto premuto. Questo non compare sullo schermo, ma può essere depositato in una variabile per futuri confronti. INEKY$ è simile a GET, ma è ancora il valore ASCII del tasto a venir riportato. Se non si preme alcun tasto entro il limite di tempo, si ottiene il valore −1. INEKY e INKEY$ ricavano i loro valori dall'ultimo carattere immesso nel *buffer* della tastiera, (quella sezione di memoria nella quale viene memorizzata la battitura). Ciò determina scarsa precisione e velocità nei giochi dove occorre premere i tasti con rapidità o tenerli abbassati: il buffer potrebbe ancora contenere il carattere battuto in precedenza! Esiste comunque una versione alternativa di questi comandi, che permette un'ispezione della tastiera, anziché del buffer.

Si può far seguire INKEY da un numero negativo tra parentesi: in questo caso, il numero non è un limite di tempo, ma rappresenta invece il codice del tasto che si vuol controllare: se volessimo verificare la pressione del tasto [N], useremmo INKEY (−86). Una lista di tutti i codici è riportata nel manuale. INKEY con un numero negativo esamina tutti i tasti premuti, anche in caso di pressioni simultanee: questa tecnica risulta molto utile nella grafica o nei giochi dove servono, ad esempio, movimenti diagonali tramite l'uso di due tasti. Il programma seguente ne è una dimostrazione (lo si confronti con il precedente, che usava GET$, e si vedrà quanto risultano più omogenee le linee tracciate):

```
10 MODE 5
20 X=500:Y=500
30 MOVE X,Y
40 REPEAT
50 IF INKEY(−58) THEN Y=Y+4
60 IF INKEY(−42) THEN Y=Y−4
70 IF INKEY(−26) THEN X=X−4
80 IF INKEY(−122) THEN X=X+4
90 DRAW X,Y
100 UNTIL INKEY(−99)
```

Con i tasti per il controllo del cursore si disegna e con la barra spaziatrice ci si ferma. Per adesso, il programma disegna una linea continua e non c'è modo di interromperla ma, aggiungendo queste quattro linee, si potrà scegliere se disegnare in nero, in modo invisibile, come pure in rosso, giallo o bianco. Per i colori si premano [B], [R], [Y] o [W].

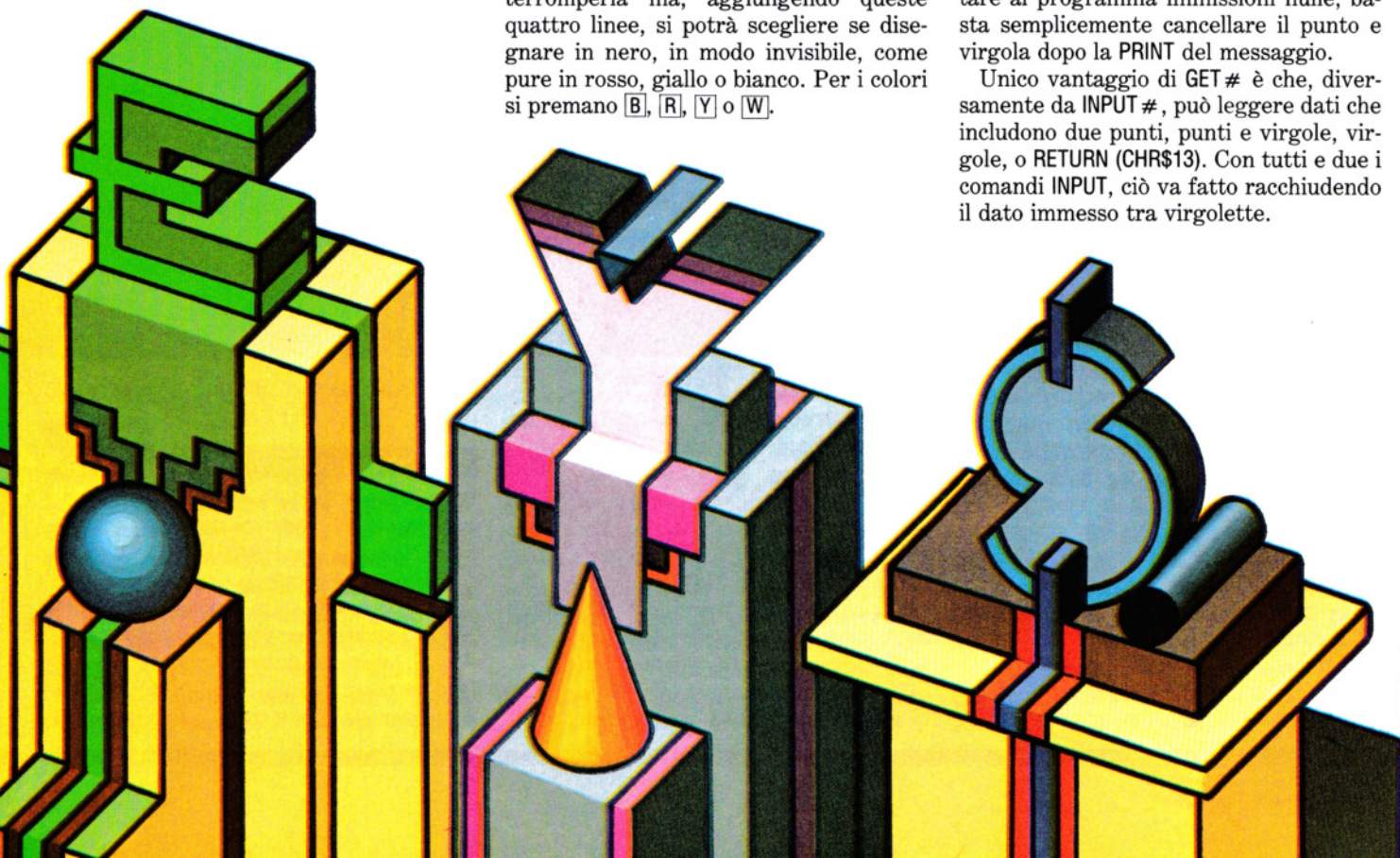
```
42 IF INKEY(−101) THEN GCOL 0,0
44 IF INKEY(−52) THEN GCOL 0,1
46 IF INKEY(−69) THEN GCOL 0,2
48 IF INKEY(−34) THEN GCOL 0,3
```

Sui computer Commodore ci sono due istruzioni, oltre alle solite INPUT e GET, per l'immissione di informazioni. INPUT# e GET# sono comandi di *input/output* normalmente usati per leggere o scrivere dati da una periferica o da un file. GET# legge un solo carattere alla volta, mentre INPUT# legge dati sotto forma di variabili, lunghe fino a 80 caratteri. Di questi due comandi, INPUT# è forse il più utile. Può venire incorporato come parte di una routine di INPUT "a prova di bomba".

```
100 OPEN 1,0: PRINT
   "COMMENTO/MESSAGGIO";:INPUT #1,
   A$: PRINT: CLOSE1
```

I comandi OPEN e CLOSE possono essere attivi, se necessario, durante l'intera esecuzione del programma. Tuttavia, non possono essere adoperati singolarmente: sono comandi complementari.

Proviamo a lanciare questo programma di una sola linea e poi ad uscirne simulando una possibile immissione di INPUT a caso. È quasi impossibile! In questo caso, l'unica via d'uscita è premere simultaneamente [RUN/STOP] e [RESTORE], una combinazione di tasti che raramente capita di premere per sbaglio. Volendo fare accettare al programma immissioni nulle, basta semplicemente cancellare il punto e virgola dopo la PRINT del messaggio.

Unico vantaggio di GET# è che, diversamente da INPUT#, può leggere dati che includono due punti, punti e virgole, virgole, o RETURN (CHR$13). Con tutti e due i comandi INPUT, ciò va fatto racchiudendo il dato immesso tra virgolette.

# SMISTIAMO LE SPESE

**Proprio come un apparecchio commerciale, anche un microcomputer può memorizzare record di dati finanziari e operare su di essi: ecco un programma che si occupa del bilancio familiare**

Tener le fila delle spese familiari ("Dove vanno a finire i soldi?") è un problema che accomuna un po' tutti. In queste pagine è presentato un programma per il bilancio familiare, adatto a tutti gli apparecchi eccetto il Vic 2Ø e lo ZX81.
Per tenere aggiornati i conti, esso va "alimentato" una volta al mese, o appena se ne ha il tempo, con la lista dettagliata delle entrate (lo stipendio, ad esempio) e delle spese (ad esempio, gli assegni emessi e i costi fissi). In qualsiasi momento, si può ottenere un'analisi di come siano stati spesi i soldi e un confronto fra entrate e uscite relativo agli ultimi dodici mesi.

Il programma è piuttosto lungo ma, una volta scritto e conservato su uno o più nastri, può durare teoricamente per sempre, o almeno finché dura il nastro magnetico.

Il programma prevede una colonna per le entrate e sette per le spese, sotto diverse intestazioni. Queste ultime si possono adattare, ovviamente, secondo le proprie necessità: è sufficiente cambiare la dicitura nella frasi DATA del programma durante la trascrizione. Le entrate, comunque, devono essere l'ultima voce e, in tutto, si devono avere otto colonne.

La memorizzazione su nastro riguarda due sezioni distinte: il programma vero e proprio e tutte le informazioni inserite, fino all'ultima immissione. Occorre, quindi, un nome diverso per ciascuno dei due file da memorizzare.

Per conservare il programma, si segua la procedura SAVE adatta al proprio apparecchio, esposta nei vari manuali alle pagine 22-25 di Input.

Per caricare il programma, basta eseguire la normale procedura LOAD, usata per i nastri dei giochi o di altri programmi. Le istruzioni per conservare e caricare i dati immessi sono spiegate in seguito. Eseguito il RUN, il menu principale offre sette opzioni:

**1** Immissione dati
**2** Visione dati
**3** Scrittura su nastro
**4** Lettura da nastro
**5** Stampa
**6** Variazione dati
**7** Fine lavoro

## IMMISSIONE DI UN DATO

Per immettere un dato, si prema 1 all'apparire del menu principale. A questo livello, non si prema ENTER o RETURN.

Il computer richiede, in successione, le seguenti informazioni: Data; Voce; Importo; Categoria (la categoria già scelta e digitata nella frase DATA).

Si scrivano le informazioni nell'ordine dato, premendo ENTER o RETURN (secondo il computer usato) dopo ogni voce.

Per tornare al menu principale, una volta completate tutte le immissioni, si attenda la richiesta di una nuova data, poi si prema ENTER o RETURN.

## VISIONE DI UN DATO

Per visionare un dato, o una serie di dati, si prema 2 all'apparire del menu principale. Non si prema ENTER o RETURN.

Il computer visualizza una tabella con le varie categorie, sette per le spese, una per le entrate. Per scegliere una categoria, si prema il numero corrispondente (ancora non si prema ENTER o RETURN) e il computer elencherà tutte le voci presenti

in quella categoria con, alla fine, il totale aggiornato.

Sullo Spectrum, può apparire sullo schermo la domanda "scroll?", se non c'è abbastanza spazio per visualizzare tutti i dati simultaneamente. Non si prema N, perché occorre listare tutte le voci.

Quando si è finito, si prema ENTER o RETURN per tornare al menu principale.

Scegliendo l'opzione 8, si ottiene non soltanto il totale delle entrate, ma anche il totale per tutte le spese, oltre al saldo tra entrate ed uscite.

## VARIAZIONE DI UN DATO

L'opzione 6 serve allo scopo di modificare un dato: il computer visualizza una lista di tutte le immissioni fatte, senza badare alla categoria.

Ci si può muovere in avanti e indietro lungo la lista seguendo le istruzioni che appaiono sullo schermo e il computer ci informa anche su come apportare correzioni ai dati.

Premuto ENTER o RETURN a variazione eseguita, il computer ritorna automaticamente al menu principale. Per apportare un'ulteriore modifica, è necessario selezionare nuovamente l'opzione 6.

## L'OPZIONE STAMPANTE

Il comando per ottenere l'uscita su stampante è della massima semplicità.

Premendo l'opzione 5 del menu principale (senza ENTER o RETURN), il computer chiede di premere S per l'uso della stampante, N in caso contrario. Se si preme una S, viene riproposto il menu principale e, fino al ritorno all'opzione 5 e al disinserimento della stampante, tutte le informazioni normalmente visualizzate sullo schermo con l'opzione 2 verranno invece stampate.

Si faccia attenzione a non premere S, se la stampante non è collegata. In questo caso, lo Spectrum ignora l'istruzione, ma sugli altri computer si perderebbero tutte le informazioni immesse fino a quel momento.

## SAVE E LOAD

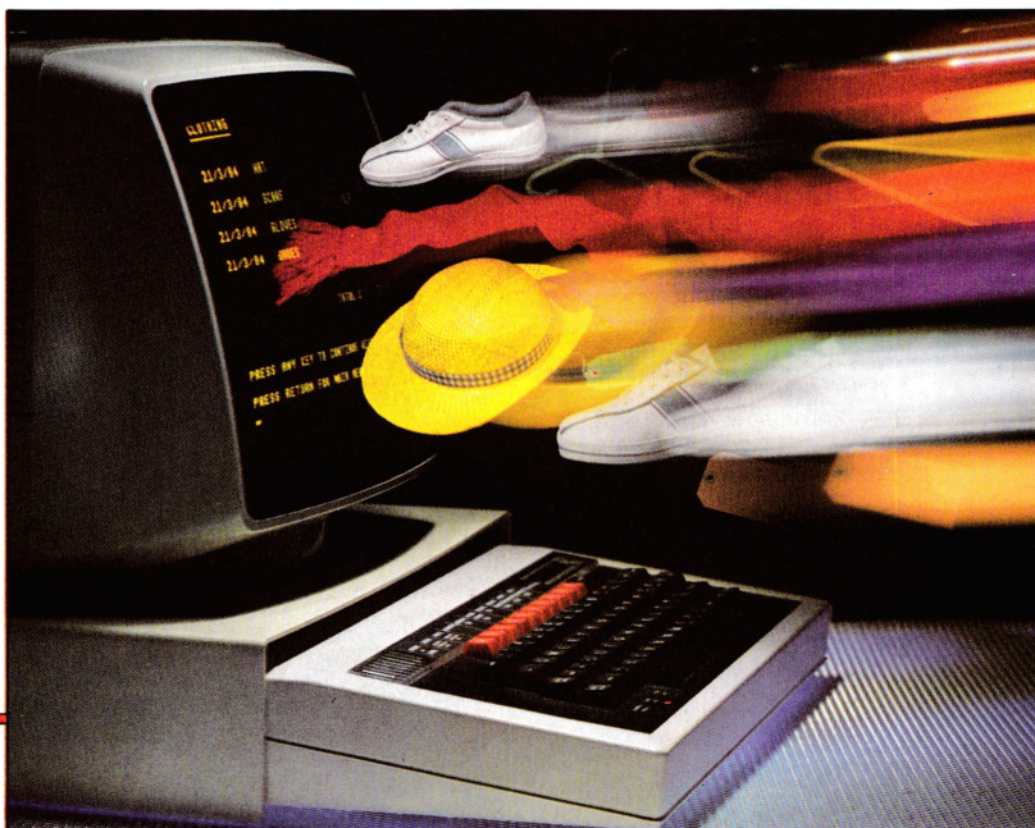
La memorizzazione dei dati inseriti avviene secondo il seguente procedimento.

Per conservare su nastro i dati immessi si prema 3 (senza RETURN o ENTER), poi si scriva un nome per il file, "BILANCIO", per esempio. Quindi, si prema RETURN o ENTER e il tasto di registrazione sul registratore. Quando i dati saranno stati conservati, verrà riproposto il menu principale dove si potrà scegliere l'opzione 7 per uscire dal programma.

Per rileggere le informazioni precedentemente conservate, si sceglie l'opzione 4 del menu principale. Dopo aver immesso il nome del file che intendiamo leggere e aver premuto RETURN o ENTER, si avvii il registratore. Una volta terminato il trasferimento, il computer riproporrà il menu principale.

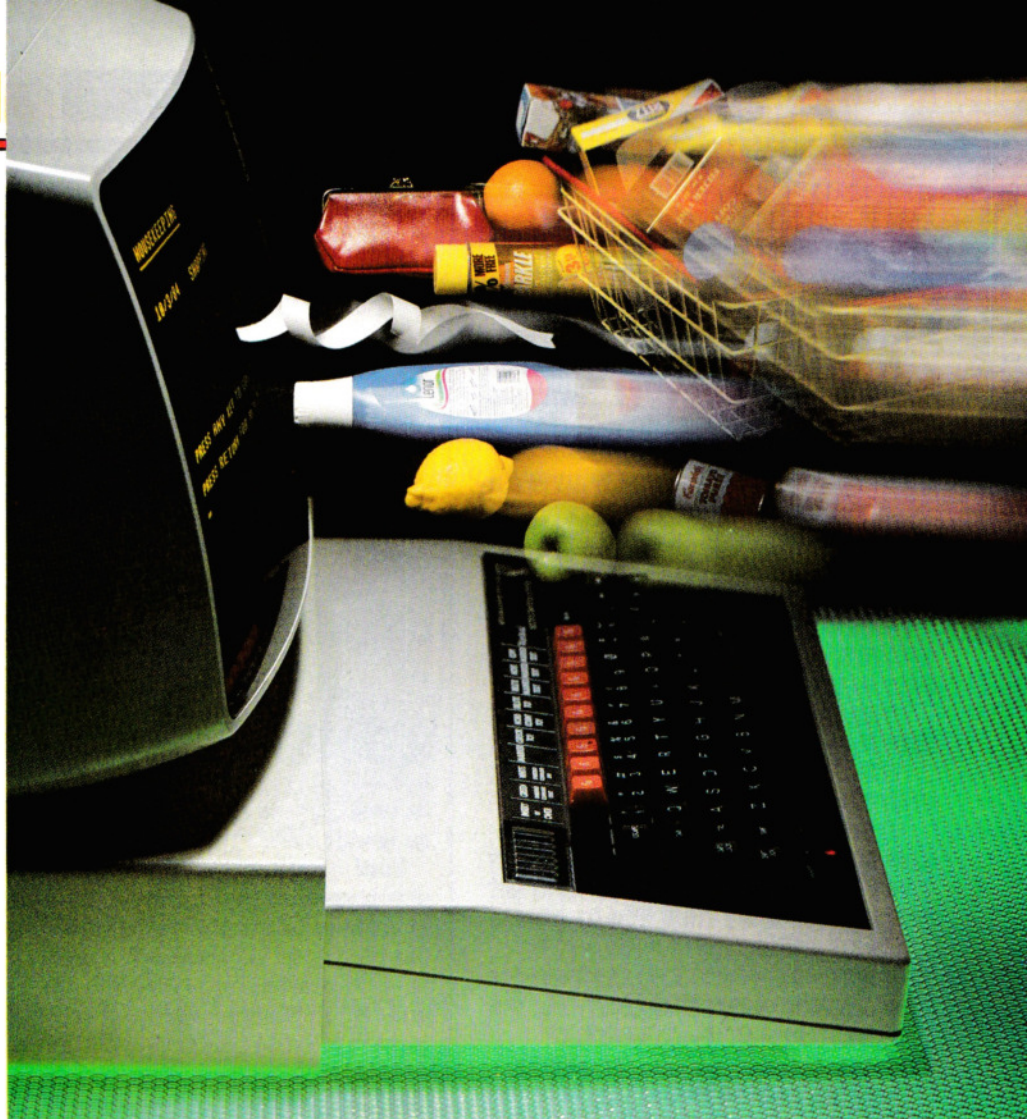
```
10 MODE6
15 * TAPE
20 *OPT1,1
30 *OPT2,1
40 @%=&2020A:N=0:W=3:VDU14:PAG=
   0:SPESE=0:DIM A$(300),A(300), D$
   (300),K$(7)
50 FOR T=0 TO 7:READ K$(T): NEXT T
60 PROCMENU
70 IF A=1 THEN PROCIMM
80 IF A=2 THEN PROCVIS
90 IF A=3 THEN PROCSAVE
100 IF A=4 THEN PROCLOAD
110 IF A=5 THEN PROCSTAMPA
120 IF A=6 THEN PROCMODIF
130 IF A<>7 THEN 60
140 PRINT"SEI SICURO (S/N)?": G=GET
   AND &5F:IF G<>83 THEN 60
150 MODE6:END
160 DEF PROCIMM
170 Z=0:CLS
180 IF N>299 THEN PRINT"
   "MEMORIA PIENA":SOUND1,-15,100,5:G
   =INKEY(200):ENDPROC
190 PRINT'"Premere RETURN al posto della
   DATA per il MENU"
200 PRINT'"DATA□□□□□□DESCR□□□
   □□□□□□□□□LIRE□□□CAT"
210 VDU28,0,24,39,4
220 INPUT TAB(0,Z)D$(N+1):IF D$(N+1)=
   "" THEN 350
230 INPUT TAB(10,Z)A$(N+1):INPUT TAB
   (26,Z)A(N+1):INPUT TAB(35,Z)CA$
240 D$(N+1)=LEFT$(D$(N+1),8):A$
   (N+1)=LEFT$(A$(N+1),16)
250 GOTO 270
260 PRINT TAB(35,Z)"□□□□□":
   INPUT TAB(35,Z)CA$
270 X=0:FOR T=0 TO 7:IF INSTR(K$(T),
   CA$)=1 THEN X=X+1:Y=T
280 NEXT
290 IF X<>1 THEN 260
300 A$(N+1)=CHR$(Y)+A$(N+1)
310 IF Y=7 THEN PAG=PAG+A(N+
   1) ELSE SPESE=SPESE+A(N+1)
320 Z=Z+1:N=N+1
330 IF Z>19 THEN Z=0:CLS
```

```
340 GOTO220
350 VDU28,0,24,39,0:ENDPROC
360 DEF PROCSHOWCAT
370 CLS:SUM=0:VDU W
380 PRINT'K$(C) 'STRING$(LEN(K$(C)),
    CHR$(224))
390 VDU28,0,24,39,3
400 FOR T=1 TO N
410 IF N=0 THEN 460
420 S$=RIGHT$(A$(T),1)
430 IF ASC(LEFT$(A$(T),1))<>
    C THEN 460
440 PRINT'D$(T)TAB(10)RIGHT$(A$(T),LEN
    (A$(T))-1)TAB(29),A(T);
450 SUM=SUM+A(T)
460 NEXT
470 PRINTTAB(32)"————————"
480 PRINTTAB(22)"TOTALE□:",SUM
490 IF C<>7 THEN 520
500 PRINT'"TOTALE DELLE SPESE□:";SPE
510 PRINT''"IL SALDO È□:"; PAG-SPESE
520 VDU1,10,1,10
530 VDU 3
540 PRINT''"PREMERE QUALSIASI TASTO
    PER CONTINUARE"''"OPPURE RETURN
    PER TORNARE AL MENU"
550 G=GET:VDU28,0,24,39,0:ENDPROC
560 DEF PROCVIEW
570 CLS:PRINT
580 FOR T=0 TO 7:PRINT'TAB(10);STR$
    (T+1);"□□";K$(T):NEXT
590 PRINT''"NUMERO DELLA CATEGORIA?";
600 C=GET-49
610 IF C=-36 THEN ENDPROC
620 IF C<0 OR C>7 THEN 600
630 PROCSHOWCAT
640 IF G=13 THEN 650 ELSE 570
650 ENDPROC
660 DEF PROCMENU
670 CLS
680 PRINTTAB(10,2)"MENU PRINCIPALE"
690 PRINTTAB(10,5)"1:— Immissione"
700 PRINTTAB(10,7)"2:— Visione"
710 PRINTTAB(10,9)"3:— Scrittura su
    nastro"
720 PRINTTAB(10,11)"4:— Lettura da
    nastro"
730 PRINTTAB(10,13)"5:— Stampa dei dati"
740 PRINTTAB(10,15)"6:— Modifica"
750 PRINTTAB(10,17)"7:— Fine lavoro"
760 PRINTTAB(10,20)"QUALE SCEGLI"
770 A=GET-48
780 IF A<1 OR A>7 THEN 770
790 ENDPROC
800 DEF PROCSAVE
810 INPUT "NOME DEL FILE",D$:IF D$=""
    THEN ENDPROC
820 IF D$="" THEN ENDPROC
830 H=OPENOUT(D$):PRINT
    "REGISTRAZIONE DELLE INFORMAZIONI":
    PRINT#H,N
840 FOR T=1 TO N:PRINT#H,D$(T),A$(T),A
    (T):NEXT:CLOSE#H:ENDPROC
850 DEF PROCLOAD
860 INPUT"NOME DEL FILE",D$
870 PRINT"AVVIARE IL REGISTRATORE"
880 H=OPENIN(D$):INPUT#H,N
890 FOR T=1 TO N:INPUT#H,D$(T),A$(T),A
    (T)
900 IF ASC(A$(T))=7 THEN PAG=PAG+A
    (T) ELSE SPESE=SPESE+A(T)
910 NEXT:CLOSE#H:ENDPROC
920 DEF PROCPRINT
930 PRINT"STAMPANTE (S/N)"
940 G=GET AND &5F:IF G=83 THEN W
    =2:GOTO 960
950 IF G<>78 THEN 940 ELSE W=3
960 ENDPROC
970 DEF PROCCHANGE
980 CLS:T=1
990 IF N=0 THEN ENDPROC
1000 REPEAT
1010 CLS:PRINT''"NUMERO OPERAZIONE□";
    STR$(T)'D$(T),RIGHT$(A$(T),LEN(A$(T))-
    1)'"Lit.";A(T),K$(ASC(A$(T)))"
1020 PRINT''"',' PER ANDARE INDIETRO"' "'.'
    PER ANDARE AVANTI"'"SPAZIO PER
    CAMBIO OPERAZIONE"
1030 A$=GET$
1040 IF A$="," THEN T=T-1: IF T<1
    THEN T=1
1050 IF A$="." THEN T=T+1:IF T>N
    THEN T=N
1060 UNTIL (A$="□" OR A$=CHR$(13))
1070 IF A$=CHR$(13) THEN ENDPROC
1080 E=T:PRINT'"MODIFICA
    ALL'OPERAZIONE"
1090 CA$=CHR$(ASC(A$(E)))
1100 IF ASC(A$(E))=7 THEN PAG=PAG-
    A(E) ELSE SPESE=SPESE-A(E)
1110 INPUT "DATA□", Q$: IF Q$ <>
    "" THEN D$ (E) = Q$
1120 INPUT"OPER.□",Q$:IF Q$<>
    "" THEN A$(E)=Q$ ELSE A$(E)=
    RIGHT$(A$(E),LEN(A$(E))-1)
1130 INPUT"IMPORTO□",Q$:IF Q$<>
    "" THEN A(E)=EVAL(Q$)
1140 INPUT"CATEGORIA□",Q$:IF Q$<>
    "" THEN CA$=Q$
1150 GOTO 1170
1160 INPUT"REINSERIRE LA CATEGORIA",
    CA$
1170 X=0:FOR T=0 TO 7
1180 IF INSTR (K$ (T),CA$)=1 THEN
    X=X+1:Y=T
1190 NEXT T
1200 IF X<>1 THEN 1160
1210 A$(E)=CHR$(Y)+A$(E)
1220 IF Y=7 THEN PAG=PAG+A
    (E) ELSE SPESE=SPESE+A(E)
1230 PRINT"CORREZIONE ESEGUITA":G=
    INKEY(200)
1240 ENDPROC
1250 DATA SPESE VARIE, DIVERTIMENTI,
    AFFITTO, ABBIGLIAMENTO, AUTO,
    VACANZE, LIBRI, ENTRATE
```

```
50 LET mn=200: IF PEEK 23733=127
   THEN LET mn=100
100 DIM C$(8,16): DIM a(mn): DIM A$(mn,
   23)
110 LET u=0: LET v=1
120 FOR n=vTO 8: READ c$(n): NEXT n
130 POKE 23658,8
140 LET k$=".00": FOR n=v TO 7: LET k$
   =k$+CHR$ 8: NEXT n
190 LET p=2: LET tt=u: LET cr=u
200 CLS: PRINT BRIGHT v; PAPER 2; INK
   6;AT 2,6;
   "□□□MENU□□□PRINCIPALE□□□"
210 PRINT BRIGHT v; PAPER 7;AT 5,6;"□1:
   —□IMMISSIONE□□";AT 7,6;"□2: —□
   VISIONE□□□";AT 9,6;"□3: —□
   SCRITTURA SU NASTRO□□";AT 11,6;"□
   4: —□LETTURA DA NASTRO□□";
   AT 13,6;"□5: —□STAMPA DEI DATI□
   □";AT 15,6;"□6: —□MODIFICA□□□";
   AT 17,6;"□7: —□FINE LAVORO□□□"
220 PRINT INK 3; FLASH v; BRIGHT v;AT
   20,6;"□—□QUALE SCEGLI□—□
230 IF INKEY$="" THEN GOTO 230
240 LET z$=INKEY$: IF z$<"1" OR z$
   >"7" THEN GOTO 230
250 CLS: GOSUB 1000*VAL z$
260 GOTO 200
1000 LET c=u
1005 LET c=c+v: IF c=mn+v THEN
   RETURN
1006 IF a$(c,v)="□" THEN GOTO 1010
1007 GOTO 1005
1010 PRINT AT u,u; BRIGHT v; PAPER 2;
   IMK7;"□□DATA□□□□□□OPER.□□
   □□□□IMPORTO□CAT"
1015 IF c=mn+v THEN RETURN
1020 INPUT "Immettere la data□"; LINE
   a$(c,2 TO 9): IF a$(c,2)="□" THEN
   RETURN
1030 PRINT TAB u;a$(c,2 TO 9);
1040 INPUT "Operazione□"; LINE a$(c,10 TO
   23): IF a$(c,10)="□" THEN GOTO 1040
1050 PRINT TAB 9;a$(c,10 TO 21);
1060 INPUT "Importo□";a(c): IF a(c)=u
   THEN GOTO 1060
1070 LET vv=a(c)*100: LET v$=STR$ vv:
   PRINT TAB 27—LEN v$;a(c);
1080 INPUT "Categoria□"; LINE f$: IF f$
   ="" THEN GOTO 1080
1090 FOR n=v TO 8: IF f$=c$(n,v TO LEN
   f$) THEN GOTO 1130
1100 NEXT n: GOTO 1080
1130 IF n=8 THEN LET cr=cr+a(c)
1140 IF n<>8 THEN LET tt=tt+a(c)
1150 PRINT TAB 29;c$(n,v TO 3)
1160 LET a$(c,v)=CHR$ (48+n)
1200 LET c=c+v: GOTO 1015
2000 FOR n=v TO 8: PRINT PAPER v; INK
   7;AT n*2,6;"□";n;": —□;c$(n): NEXT n
2010 PRINT FLASH v; INK 2;AT 19,3;
   "□Quale categoria (da 1 a 8)□"
2020 IF INKEY$="" THEN GOTO 2020
2030 LET z$=INKEY$: IF z$<"1" OR z$
   >"8" THEN GOTO 2020
2040 LET t=u: LET c=u
2050 CLS: PRINT #p; PAPER 6; BRIGHT
   v;TAB 10;c$(VAL z$); TAB 31; "□"
2055 LET c=c+v: IF c=mn THEN GOTO
   2500
2060 IF a$(c,v)="□" THEN GOTO 2500
2070 IF a$(c,v)<>z$ THEN GOTO 2055
2080 PRINT #p;a$(c,2 TO 9); TAB 10;
   a$(c,10 TO 23);
2090 LET am=a(c)*100: LET n$=STR$ am:
   PRINT #p;TAB 29;k$;TAB 31—LEN
   n$;a(c)
2100 LET t=t+a(c)
2110 GOTO 2055
2500 PRINT #p;TAB 25;"——————
   —": LET tx=t*100:LET n$=
   STR$ tx: PRINT #p;TAB 12;"TOTAL:—
   □";TAB 29;k$;TAB 31—LEN n$;t
2510 IF z$<>"8" THEN GOTO 2590
2520 LET tz=tt*100:L LET n$=STR$ tz:
   PRINT'#p;"TOTALE DELLE SPESE:
   —□";TAB 29;k$;TAB 31—LEN n$;tt
2530 LET ba=(t—tt)*100: LET n$=STR$
   ba: PRINT'#p;TAB 10; "SALDO:—□"TAB
   29;k$;TAB 31—LEN n$;ba/100
2590 PRINT PAPER 2; INK 7'
   "□□□Premere un tasto per
   continuare□□"
2600 PAUSE u: IF PEEK 23560=13 THEN
   RETURN
2610 CLS: GOTO 2000
3000 GOSUB 8000: IF re=v THEN RETURN
3010 PRINT PAPER 6;AT 10,u;"□Nome per il
   file□": INPUT LINE w$: IF LEN w$>10
   OR LEN w$<v THEN GOTO 3010
3020 CLS: SAVE w$ DATA a(): SAVE w$
   DATA a$(): RETURN
4000 GOSUB 8000: IF re=v THEN RETURN
4010 PRINT BRIGHT v;AT 10,u;"Nome del file
   da leggere": INPUT LINE w$: IF LEN w$
   >10 THEN GOTO 4010
4020 PRINT PAPER 3; INK 7;AT
   10,u;"□□□Avviare il registratore□□□"
4030 LOAD w$ DATA a(): LOAD w$ DATA
   a$()
4040 LET cr=u: LET tt=u: FOR n=v TO
   mn: IF a$(n,v)="8" THEN LET cr=cr
   +a(n)
4050 IF a$(n,1)<>"8" THEN LET tt=tt
   +a(n)
4060 NEXT n: RETURN
5000 PRINT BRIGHT v;AT 10,u; "□Vuoi fare
   una stampa (S/N)?□"
5010 PAUSE u: IF INKEY$="" THEN GOTO
   5010
5020 LET z$=INKEY$
5030 IF z$="N" THEN LET p=2: RETURN
5040 IF z$="S" THEN LET p=3: RETURN
5050 GOTO 5010
6000 LET c=v: IF a(c)=u THEN RETURN
```

```
6010 PRINT AT u,u; BRIGHT v; PAPER (VAL
    a$(c,v))—v; INK 9;"□Numero□";c,c$(VAL
    a$(c,v))
6015 PRINT PAPER 2; INK 7;'"DATA□□□
    □□□OPER□□□□□□□□□□□
    IMPORTO□":PRINT'a$(c,2 TO 9);TAB 10;
    a$(c, 10 TO 23);
6020 LET am=a(c)*100: LET n$=STR$ am:
    PRINT TAB 29;k$; TAB 31—LEN n$;a(c)
6030 PRINT PAPER 3; INK 7;AT 20,u;"□A□
    —□Avanti□□□□I□—□Indietro□□
    □□□□□EDIT per modificare un record
    □□□□□□□"
6040 PAUSE u
6050 IF INKEY$="I" AND c>v THEN LET c
    =c—v: GOTO 6010
6060 IF INKEY$="A" AND c< >mn THEN
    LET c=c+v
6070 IF a(c)=u THEN LET c=c—v
6080 IF PEEK 23560=7 THEN GOTO 6100
6090 GOTO 6010
6100 INPUT BRIGHT v; "Nuova data□"; LINE
    a$(c,2 TO 9): IF a$(c,2)="□" THEN
    GOTO 6100
6110 PRINT AT 5,u;a$(b,2 TO 9)
6120 INPUT BRIGHT v;"Nuova operazione□";
    LINE a$(c,10 TO 23): IF a$(c,10)="□"
    THEN GOTO 6120
6130 PRINT AT 5,10;a$(c,10 TO 23)
6135 IF a$(c,v)="8" THEN LET cr=cr—a(c)
6136 IF a$(c,v)< >"8" THEN LET tt=tt
    —a(c)
6140 INPUT BRIGHT v;"Nuovo
    importo□";a(c): IF a(c)=u THEN GOTO
    6140
6150 LET am=a(c)*100: LET n$=STR$ am:
    PRINT AT 5,29;k$;TAB 31—LEN n$;a(c)
6160 INPUT BRIGHT v;"Nuova categoria□";
    LINE f$: IF f$="" THEN GOTO 6160
6170 FOR n=v TO 8: IF f$=c$(n,v TO LEN
    f$) THEN GOTO 6190
6180 NEXT n: GOTO 6160
6190 LET a$(c,v)=CHR$ (48+n)
6200 IF n=8 THEN LET cr=cr+a(c)
6210 IF n<8 THEN LET tt=tt+a(c)
6220 RETURN
7000 GOSUB 8000: IF re=v THEN RETURN
7010 RANDOMIZE USR u
8000 PRINT PAPER 4;AT 10,9;"□Sei
    sicuro?□"
8010 PAUSE u: LET re=u: IF INKEY$<
    >"S" THEN LET re=v
8020 RETURN
9000 DATA "SPESE VARIE",
    "DIVERTIMENTI", "AFFITTO",
    "ABBIGLIAMENTO", "AUTO", "VACANZE",
    "LIBRI", "ENTRATE"
```

Sul Tandy, usare 247 invece di 223 nelle linee 6040, 6050 e 6060.

```
10 PMODE0:PCLEAR10000
20 DIM TE$(200),AM(200),DA$(200),CT$(8),
    CA(200)
30 FOR N=1 TO 8:READ CT$(N):NEXT
40 DATA SPESE VARIE,DIVERTIMENTO,
AFFITTO,ABBIGLIAMENTO,AUTO,VACANZE,
LIBRI,ENTRATE
50 U1$="#######.##":U2$=
    "####.##"
60 CLS4:PRINT@11,"menu principale";:
    PRINT@70,"1:—IMMISSIONE□□□□□
    □";:PRINT@134,"2:—□VISIONE□□□
    □";:PRINT@198,"3:—□
    SCRITTURA SU NASTRO□□□";
70 PRINT@262,"4:—LETTURA DA NASTRO
    □□";:PRINT@326,"5:—□
    STAMPA DEI DATI□□";:PRINT@390,"6:
    —□VARIAZIONE□";:PRINT@454,"7:—□
    FINE LAVORO□□□□";
80 A$=INKEY$:IF A$<"1" OR A$>"7"
    THEN80
90 ON VAL(A$) GOSUB 1000,2000,3000,
    4000,5000,6000,7000
100 GOTO 60
1000 CLS:IF NU>200 THEN PRINT@264,
    "MEMORIA PIENA□!":PLAY"T10ABCDEFG
    P1P1":RETURN
1010 GOSUB 1160
1020 GOSUB 1250:INPUT"DATA□";DA$(NU)
1030 IF DA$(NU)="" THEN RETURN
1040 IF LEN(DA$(NU))>8 THEN 1020
1050 PRINT@L,DA$(NU)
1060 GOSUB 1250:LINEINPUT
    "OPERAZIONE□?";TE$(NU):IF LEN(TE$
    (NU))>25 THEN1060
1070 A$=LEFT$(TE$(NU),11):PRINT@L+
    15—LEN(A$)/2,A$;
1080 GOSUB 1250:INPUT"IMPORTO□";A
1090 IF A>9999.99 OR A<0 THEN 1080
1100 PRINT@L+21,USING U2$;A;:AM(NU)
    =A
1110 GOSUB 1250:INPUT"CATEGORIA□";CA$
1120 GOSUB 1180:IF F=0 THEN 1110
1130 CA(NU)=NM:PRINT@L+29,LEFT$
    (CT$(CA(NU)),3);
1140 IF NM< >8 LENGHT GT=GT+A
1150 NU=NU+1:L=L+32:IF L=448
    THEN 1000 ELSE 1020
1160 L=64:PRINT@2,"data"TAB(13)"oper"
    TAB(22)"importo"TAB(29)"cat";
1170 RETURN
1180 IF VAL(CA$)< >0 THEN 1230
1190 F=0:FOR N=1 TO 8
1200 IF CA$=LEFT$(CT$(N),LEN(CA$))
    THEN F=F+1:NM=N
1210 NEXT:IF F>1 THEN F=0
1220 RETURN
1230 IF VAL(CA$)>8 THEN F=0:RETURN
1240 NM=VAL(CA$):F=1:RETURN
1250 PRINT@448,"□":PRINT@449,;:RETURN
2000 CLS3:FOR N=1 TO 8
2010 PRINT@69+N*32,N;MID$(":—□"+
```

```
    CT$(N) + STRING$(12,"□"),1,20);
2020 NEXT
2030 TT = 0:PRINT@449,"QUALE
    CATEGORIA□?"
2040 A$ = INKEY$:IF A$ < "1" OR A$ > "8"
    THEN 2040
2050 NM = VAL(A$)
2060 IF PT = 1 THEN PRINT # − 2,CHR$(13)
    :PRINT # − 2,TAB(21 − LEN(CT$(NM))/2);
    CT$(NM):PRINT # − 2,"□□DATE"TAB
    (20)"ITEM"TAB(39)"AMOUNT"
2070 GOSUB 2280
2080 FOR NN = 0 TO NU
2090 IF CA(NN) < > NM THEN 2150
2100 IF PT = 1 THEN PRINT # − 2,USING
    F$; DA$(NN);TE$(NN);AM(NN)
2110 PRINT@L,DA$(NN);:A$ = LEFT$(TE$
    (NN),15):PRINT@L + 17 − LEN(A$)/2,A$;:
    PRINT@L + 25,USING U2$;AM(NN);:TT =
    TT + AM(NN)
2120 L = L + 32:IF (L = 448 AND NM < > 8)
    OR (L = 352 AND NM = 8) THEN PRINT@
    465, "scroll□?"; ELSE GOTO 2150
2130 A$ = INKEY$:IF A$ = "" THEN 2130
2140 GOSUB 2280
2150 NEXT:IF PT = 1 THEN PRINT # − 2,
    CHR$(13):IF NM < > 8 THEN PRINT # −
    2,TAB(28);:PRINT # − 2,USING"TOTALE□
    = " + U1$;TT
2160 PRINT@463,USING"totale□ = " + U1$;
    TT;
2170 IF NM < > 8 THEN 2250
2180 IF PT = 0 THEN 2220
2190 PRINT # − 2,TAB(21);:PRINT # − 2,
    USING"TOTALE ENTRATE□ = " + U1$;TT
2200 PRINT # − 2,TAB(16);:PRINT # − 2,
    USING"TOTALE USCITE□ = " + U1$;GT:
    PRINT # − 2,TAB(35)" − − − − − − −
    − − −"
2210 PRINT # − 2,TAB(26);:PRINT # − 2,
    USING"SALDO□ = " + U1$;TT − GT
2220 PRINT@392,USING"totale entrate□ = "
    + U1$;TT;
2230 PRINT@419,USING"totale uscite□ = "
    + U1$;GT;
2240 PRINT@491,USING"saldo□ = " + U1$;
    TT − GT;
2250 A$ = INKEY$:IF A$ = "" THEN 2250
2260 IF A$ < > CHR$(13) THEN 2000
2270 RETURN
2280 L = 64:CLS NM:PRINT@(33 − LEN(CT$
    (NM)))/2,CT$(NM);
2290 PRINT@34,"data";:PRINT@46,"oper";:
    PRINT@57,"importo";
2300 FOR N = 32 TO 416 STEP 32
2310 POKE N + 1032,122 + NM*16:POKEN +
    1048,117 + 16*NM
2320 NEXT:RETURN
3000 CLS:MOTORON:PRINT@65,
    "RIAVVOLGERE IL NASTRO E PREMERE
    [ENTER]"
3010 A$ = INKEY$:IF A$ = "" THEN 3010
3020 MOTOROFF:PRINT@65,
    "PREPARARE IL REGISTRATORE □□□□
    □□□□□□□□ E PREMERE [ENTER]"
3030 A$ = INKEY$:IF A$ = "" THEN 3030
3040 CLS:PRINT@65,;:INPUT
    "NOME DEL FILE□";DA$
3050 OPEN "O", # − 1,DA$
3060 PRINT # − 1,NU
3070 FOR N = 0 TO NU − 1
3080 PRINT # − 1,DA$(N),TE$(N),AM(N),CA
    (N)
3090 NEXT:CLOSE # − 1:RETURN
4000 CLS:PRINT@65,;:INPUT
    "NOME DEL FILE";DA$
4010 MOTORON:PRINT@64,
    "PREPARARE IL REGISTRATORE E
    PREMERE [ENTER]"
4020 A$ = INKEY$:IF A$ = "" THEN 4020
4030 MOTOROFF:GT = 0:OPEN "I", #
    − 1,DA$
4040 PRINT@129,DA$;"□□TROVATO"
4050 INPUT # − 1,NU
4060 FOR N = 0 TO NU − 1
4070 INPUT # − 1,DA$(N),TE$(N),AM(N),CA
    (N)
4080 IF CA(N) < > 8 THEN GT = GT + AM(N)
4090 NEXT:CLOSE # − 1:RETURN
5000 CLS:PRINT@65,
    "VUOI USARE LA STAMPANTE□?□□□
    □□(S/N)";
5010 A$ = INKEY$:IF A$ < > "S" AND A$ <
    > "N" THEN 5010
5020 PRINT" OK":IF A$ = "N" THEN PT = 0:
    RETURN
5030 F$ = "%□□□□□□□□□%%□□□□
    □□□□□□□□□□□□□□□□□□□
    □%□□□" + U2$:PT = 1:RETURN
6000 IF NU = 0 THEN RETURN
6010 CLS:PRINT"□□data"TAB(12)"oper"TAB
    (22)"importo"TAB(29)"cat"
6020 PRINT@417,"PREMERE [GIÙ] PER
    ANDARE AVANTI □□□□ OPPURE [SU]
    PER ANDARE INDIETRO."
6030 PRINT@481,"PREMERE UNO SPAZIO
    PER MODIFICARE";:M = 0:GOTO6080
6040 IF PEEK (341) = 223 AND M > 0 THEN
    M = M − 1:GOTO 6080
6050 IF PEEK(342) = 223 AND M < NU − 1
    THEN M = M + 1:GOTO 6080
```

```
6060 IF PEEK(345)=223 THEN 6100
6070 GOTO 6040
6080 PRINT@64,USING"%□□□□□□%□%
     □□□□□□□□□□%####.##
     □%□□%";DA$(M);LEFT$(TE$(M),11);AM
     (M);LEFT$(CT$(CA(M)),3)
6090 GOTO 6040
6100 IF CA(M)<>8 THEN GT=GT-AM
     (M)
6110 INPUT"NUOVA DATA□";D$:IF D$
     ="" THEN 6130
6120 IF LEN(D$)>8 THEN 6110 ELSE
     DA$(M)=D$
6130 INPUT"NUOVA OPERAZIONE□";D$?:
     IF D$=""THEN 6150
6140 TE$(M)=D$
6150 INPUT"NUOVO IMPORTO□";A:IF A=0
     THEN 6170
6160 IF A<0 OR A>9999.99 THEN 6150
     ELSE AM(M)=A
6170 INPUT"NUOVA CATEGORIA□";CA$:
     IF CA$="" THEN 6200
6180 GOSUB1180:IF F=0 THEN 6170
6190 CA(M)=NM
6200 IF CA(M)<>8 THEN GT=GT+AM
     (M)
6210 RETURN
7000 CLS:PRINT@69;"SEI SICURO□
     (S/N)□?";
7010 A$=INKEY$:IF A$<> "S" AND A$
     <>"N" THEN 7010
7020 IF A$="N" THEN RETURN
```

**Il Dragon e il Tandy possono produrre caratteri minuscoli, per esempio le lettere dalla 'a' alla 'z'?**

Il Dragon può produrre caratteri minuscoli sulla stampante, ma non sullo schermo. Ma anche sullo schermo si può ottenere una differenziazione tra i caratteri maiuscoli e minuscoli. Infatti, quando si accende il computer, questo è automaticamente predisposto nel modo maiuscolo: si premano i tasti SHIFT e 0 contemporaneamente. Questa procedura sblocca il modo maiuscolo (CAPS lock), cosicché le lettere premute appaiono ora in colori invertiti, maiuscole verdi su sfondo nero.

Tutti i comandi per il computer devono comunque essere in maiuscole. La stampante riproduce una 'A' per la 'A' maiuscola e una 'a' se il carattere sullo schermo è una "A" su sfondo invertito. Per tornare al modo maiuscolo, si premano una seconda volta SHIFT e 0 assieme.

```
20 PRINT "[CLR]":POKE 53280,0:POKE 53281,0:
   DIM D$(4,400):CO=0
30 A$(1)="SPESE VARIE":A$(2)=
   "DIVERTIMENTI":A$(3)="AFFITTO"
40 A$(8)="ENTRATE":A$(4)="VESTIARIO":
   A$(5)="AUTO":A$(6)=VACANZE"
50 A$(7)="LIBRI"
60 PRINT "[CLR][π][DOWN][DOWN][DOWN]"TAB(13)"[■]□□
   □□MENU PRINCIPALE□□□□":
   PRINT TAB(13)"[↑][DOWN][DOWN]1. IMMISSIONE"
70 PRINT TAB(13)"[DOWN] 2. VISIONE":PRINT
   TAB(13)"[DOWN] 3. SCRITTURA SU NASTRO"
80 PRINT TAB(13)"[DOWN] 4. LETTURA DA
   NASTRO":PRINT TAB(13)"[DOWN] 5. STAMPA
   DEI DATI"
90 PRINT TAB(13)"[DOWN]6. VARIAZIONE"
100 PRINT TAB(13)"[DOWN]7. FINE LAVORO":
   PRINTTAB(13)"[DOWN][DOWN][π][■]□□
   QUALE SCEGLI?□"
110 GET K$:IF VAL(K$)<1 OR VAL(K$)>7
   THEN 1100
120 C$="":KK$=K$:IF K$="1" THEN
   GOTO 500
130 IF K$="2" THEN GOSUB 440: GOTO
   640
140 IF K$="3" THEN GOSUB 830: GOTO
   790
150 IF K$="4" THEN GOSUB 830: GOTO
   810
160 IF K$="5" THEN PRINT "[CLR]": GOTO
   600
170 IF K$="6" AND CO <> 0 THEN C$
   ="S":CQ=1:QQ$=D$(4,1) GOTO200
180 IF K$="7" THEN PRINT TAB(13);:INPUT
   "[CLR]□SEI SICURO□□□▌▌▌▌"; K$:IFK$
   ="S"THENEND
190 GOTO 60
200 CC=0:C1=0
210 PRINT"[CLR][DOWN][DOWN]":IFC$="S"THEN
   PRINT TAB(12)"[■]OPERAZIONE
   NUMERO"CQ
220 PRINT"[DOWN][π]"TAB(20-(LEN(A$(VAL
   (QQ$)))*.5))A$(VAL(QQ$))
230 PRINT "[DOWN][↑] ---------------
   ----------------------"
240 PRINT "□□[π]DATA[↑]□□□    □□
   □□□□□[π]OPER[↑]□□□□□□□
      □□[π]IMPORTO[↑]"
250 PRINT "--------------------
   -------------------" : SC =
   0
260 IFC$="S"THENC1=CQ:GOSUB370:GOTO
   860
270 C1=C1+1:IF D$(4,C1)=QQ$ THEN
   GOSUB 370:IF PR$="N" THEN SC=SC
   +1
280 IF SC=>10 THEN SC=0:GOSUB840:
```

**Il menu principale (questa è la versione Spectrum) offre tutte queste opzioni**

```
   PRINT "▮▮▮▮▮▮"
290 IF C1=400 OR VAL(D$(4,C1))=0 THEN
   310
300 GOTO 260
310 C1=0:FOR Z=1 TO 8:N(Z)=0:NEXT:FR
   =0
320 C1=C1+1:IF C1=>400 OR VAL(D$
   (4,C1))=0 THEN RETURN
330 IF VAL(D$(4,C1))=VAL(QQ$) THEN N
   (VAL(QQ$))=N(VAL(QQ$))+VAL(D$(3,C1)
340 IF VAL(D$(4,C1))=8 AND VAL(QQ$)=
   8 THEN320
350 IF VAL(D$(4,C1))<>8 THEN FR=FR
   +VAL(D$(3,C1)
360 GOTO 320
370 PRINT LEFT$(D$(1,C1)+"□□□□□□
   □□□□",9)"   "
380 PRINTLEFT$(D$(2,C1)+"□□□□□□
   □□□□□□□□□□□□□□",18);"   £";
390 VV$=D$(3,C1):TA=9
400 VV=VAL(VV$):IF VV-INT(VV)=
   0 THEN VV$=STR$(VV)+".00"
410 IF MID$(VV$,LEN(VV$)-1,1)=
   "." THEN VV$=VV$+"0"
420 PRINT RIGHT$("□□□□□□□□□□
   □□□□□□"+VV$,TA)
430 RETURN
440 PRINT "▮▮▮▮▮▮▮▮"TAB
   (13)"▮□□□□□CATEGORIA□□□□
   ▮▮":POKE 198,0
450 FOR Z=1 TO 8:PRINT TAB(12)Z ":□"
   A$(Z):NEXT
460 PRINT TAB(13)"▮▮▮□□
   QUALE SCEGLI□?□"
470 GET K$:IF (VAL(K$)<1 OR VAL(K$)>8)
   AND K$<>CHR$(13) THEN 470
480 IF K$=CHR$(13) AND KK$="1" THEN
   470
490 PRINT "▮":QQS=K$:RETURN
500 IF K$=CHR$(13) THEN CO=CO-1
   : GOTO 60
510 CO=CO+1:IF CO>400 THEN CO=
   400: GOTO 60
520 C1=CO:D$(1,C1)=""
530 PRINT "▮▮▮":IF C$<>"S" THEN
   PRINT "PER TORNARE AL MENU,
   PREMERE RETURN INVECE DELLA DATA"
540 INPUT "▮▮▮▮□□
   IMMETTERE LA DATA ▮" D$(1,C1):IF D$
   (1,C1)=""THEN K$=CHR$(13):
   GOTO 500
550 INPUT "▮□□OPERAZIONE ▮";D$(2,
   C1)
560 INPUT "▮▮IMPORTO ▮";D$(3,C1):
   GOSUB 440:IF QQ$<>CHR$(13)
   THEN D$(4,C1)=QQ$
580 IF C$="S" THEN QQ$=D$(4,CQ):GOTO
   200
590 GOTO500
600 PRINT "▮"TAB(5)"▮▮▮▮▮
   VUOI UNA STAMPA DEI DATI (S/N)?"
610 GET K$ : IF K$ = "S" THEN PR$="S":
   RETURN
620 IF K$="N" THEN PR$="N":RETURN
630 GOTO 600
640 IF K$=CHR$(13) THEN 60
650 IF PR$="S"THEN OPEN 4,4:CMD4
660 GOSUB 200:PRINT "▮ ----------
   ----------------------------
   - ▮"
670 VV$=STR$(N(VAL(QQ$)))
680 PRINT TAB(19)"TOTALE□:▮□";:TA=
   12:GOSUB400
690 PRINT "▮ ------------------
   ---------------------": IF
   QQ$<>"8" THEN 720
700 PRINT TAB(7)"▮▮TOTALE SPESE□:
   ▮□";:TA=12
710 VV$=STR$(FR):GOSUB400:VV$=STR$
   (N(VAL(QQ$))-FR)
720 IF QQ$="8" THEN PRINTTAB(17)"▮
   ▮SALDO□:▮□";:TA=12:GOSUB 400
730 IF PR$="S" THEN PRINT#4,CHR$(13):
   CLOSE4
740 GOSUB940:K$="2":POKE 198,0
750 GETW$:IF W$=""THEN 750
760 IF W$=CHR$(13) THEN 60
770 IF PR$="S" THEN GOSUB 440: GOTO
   640
780 GOTO120
790 OPEN1,1,1,NM$:PRINT#1,CO:FORZ=
   1 TO CO:FOR ZZ=1 TO 4:PRINT#1,D$
   (ZZ,Z)
800 NEXT ZZ,Z:CLOSE1:GOTO 60
810 OPEN1,1,0,NM$:INPUT#1,CO:FORZ=
   1 TO CO:FOR ZZ=1 TO 4:INPUT#1,D$
   (ZZ,Z)
820 NEXT ZZ,Z:CLOSE1:GOTO 60
830 INPUT "▮▮▮NOME DEL FILE ▮";
   NM$:PRINT"▮":RETURN
840 PRINT TAB(11)"▮▮(▮
   PREMI UN TASTO PER CONTINUARE) ▮"
   :POKE198,0:WAIT198,1:PRINT"
   ▮▮▮▮▮▮▮";
850 FOR Z=1 TO 14:PRINT□□□□□□□
   □□□□□□□□□□□□□□□□□□
   □□□□□□□□□□□□□□□":NEXT:
   RETURN
860 PRINT"▮▮(,)=INDIETRO (.)=AVANTI
   □□□□□□□□□□□▮SPAZIO=
   MODIFICA"
870 GET P$:IF P$="" THEN 870
880 IF P$=CHR$(13) THEN 60
890 IF P$="□" THEN 530
900 IF P$="." THEN CQ=CQ+1:IFCQ>
   COTHENCQ=CO
910 IF P$="," THEN CQ=CQ-1:IFCQ<
   1THENCQ=1
920 IF P$=","ORP$="."THEN QQ$=D$(4,
   CQ):GOTO200
930 GOTO 870
940 PRINT "▮▮ PREMERE UN TASTO
   QUALSIASI PER CONTINUARE □□▮
   OPPURE RETURN PER TORNARE AL
   MENU"
950 RETURN
```

# NEMICI MORTALI E ALIENI

**Dagli Space Invaders agli ultimi giochi 'arcade', la sfida si accresce con avversari sempre nuovi e diversi. Ecco come creare questi giochi e inserirli in una routine di gioco completa**

L'aspetto dei giochi migliora di molto sfruttando la grafica ad alta risoluzione di ogni particolare apparecchio. I programmi che usano la grafica ad alta risoluzione sono più difficili di quelli che usano soltanto le lettere della tastiera, ma i risultati meritano la fatica in più.

Molti giochi tipo 'arcade' si fondano su nemici, alieni o avversari che sparano a raffica, invece di aspettare pacificamente di venire distrutti. Presentiamo perciò un gioco chiamato Stazione Spaziale, adatto a tutti i computer, tranne lo ZX81 e il Vic 2Ø, che insegna come programmare i movimenti casuali di un "alieno" sullo schermo, e come farlo sparare contro l'obiettivo costituito dalla stazione spaziale.

Per parare i missili sparati dall'alieno, il giocatore possiede quattro scudi protettivi che però non possono essere tenuti alzati per tutto il tempo, dato che l'energia necessaria per alimentarli è limitata.

Per rendere il gioco più difficile, il programma non prevede soltanto che l'alieno si muova casualmente, ma anche che scompaia nell'*iperspazio*, per riapparire in una posizione del tutto diversa.

Così com'è senza un conteggio del tempo e dei punti, il gioco non è del tutto completo, ma a ciò si rimedia applicando i metodi esposti alle pagine 97-103.

A differenza dell'alieno, che è simile a quelli dei giochi in vendita, la stazione spaziale è soltanto un contorno. Volendo, si può ridisegnare la stazione usando la grafica definita dall'utente (descritta a pagina 138) o, per i possessori di Commodore, usando gli sprite descritti a pagina 151. Si dovrà in ogni caso mantenersi entro l'area usata dalla nostra stazione spaziale, altrimenti questa si sovrapporrebbe agli schermi difensivi, comportando così numerose revisioni a tutto il programma.

Ecco la prima parte del gioco:

```
10 PCLEAR4:PMODE4,1:PCLS
15 SCREEN 1,1
20 DIM AL(6),BL(6),BO(4)
30 DEFFNZ(X)=SGN(X)*SQR(V*V*X*X/((127
   -AX)*(127-AX)+(95-AY)*(95-AY)))
40 LET PW=250
```

```
50 FORI=ØTO7:READA:POKE1536+I*32,A:
   NEXTI
60 GET (Ø,Ø)—(7,7),AL,G
65 GOTO 65
25Ø DATA 24,126,9Ø,126,126,195,129,129
```

Eseguendolo, sullo schermo appare l'alieno. La grafica ad alta risoluzione è predisposta dalla linea 1Ø. A questo livello, la linea 15 consente di osservare il funzionamento del programma sullo schermo, ma più avanti, quando il gioco sarà completato, la linea 15 sarà eliminata.

Le matrici che generano l'alieno, il missile e uno spazio vuoto, sono dimensionate dalla linea 2Ø. La linea 3Ø usa un'istruzione BASIC mai incontrata prima: DEFFN, abbreviazione di 'definizione di funzione'. Questo metodo consente di usare lunghe espressioni matematiche, evitando di riscriverle più volte in un programma. L'espressione matematica alla linea 3Ø viene richiamata, per muovere il missile in diagonale, semplicemente usando il nome FNZ. La linea 4Ø assegna il limite della riserva energetica.

La linea 5Ø disegna l'alieno sullo schermo leggendo i dati della linea 25Ø e depositandoli, mediante una POKE, in alto a sinistra sullo schermo. Il Dragon memorizza l'immagine dell'alieno, definita nella matrice AL, grazie alla GET nella linea 6Ø.

Anche la linea 65 è una linea provvisoria: ha solamente la funzione di mantenere lo schermo abilitato e in seguito sarà eliminata.

## IL DISEGNO DEL MISSILE

Si aggiungano queste linee e si dia un RUN:

```
70 FOR J=Ø TO 4:READ A:POKE 1536+J*
   32,A:NEXTJ
80 GET(Ø,Ø)—(4,4),BO,G
85 GOTO 85
26Ø DATA 32,112,248,112,32
```

Il missile in dotazione all'alieno è inserito, mediante POKE, in alto a sinistra sullo schermo. Non ha importanza se viene posto sopra l'alieno e neppure che parte di questo resti alla sua sinistra: la linea 8Ø si limita a registrare l'area occupata dal missile e non ciò che vi si trova intorno.

## IL DISEGNO DELLA STAZIONE

Si cancelli la linea 85 digitando 85 [ENTER], si aggiungano queste linee e si dia un RUN:

```
90 LET AX=RND(248)—1: LET AY=RND
   (178)+5
100 PCLS
110 CIRCLE(127,95),12,5:DRAW"BM127,95;C
   5S48NUNLNDNR"
115 GOTO 115
```

La linea 9Ø assegna all'alieno una posizione di partenza casuale. La 1ØØ ripulisce lo schermo prima che la 11Ø disegni la stazione spaziale. Il comando DRAW, al termine della linea, disegna una croce sulla stazione: la spiegazione di quest'ultimo comando è prevista più avanti; per ora lo si interpreti come una successione di istruzioni LINE, già spiegate.

## IL QUADRANTE PER L'ENERGIA

Si cancelli la linea 115 come per le 65 e 85. Aggiungendo queste linee, appariranno ulteriori dettagli grafici:

```
120 DRAW"BM131,87;S4D5BD6BLR3D2L3D2
   R3BL12R3U2NL3U2NL3BU6U5G4R3"
130 DRAW"BM5,1;L4D2NR4D2BE4BR2D4R3
   U4BR5L3D2NR3D2R3BE4D4R3"
14ØLINE(25,1)—(PW,5),PSET,BF
145 GOTO 145
```

La linea 12Ø disegna sulla stazione i numeri che corrispondono agli schermi. La linea 13Ø visualizza la parola CARBURANTE. Purtroppo il Dragon non visualizza i comuni caratteri della tastiera sullo schermo ad alta risoluzione, perciò le lettere andranno disegnate.

Il quadrante per la riserva energetica degli schermi è visualizzato dalla linea 14Ø. LINE utilizza un metodo rapido per disegnare rettangoli: viene tracciata una linea dall'angolo a sinistra in alto a quello a destra in basso. PSET fa disegnare al Dragon la linea color nocciola. BF serve per colorare il rettangolo, con lo stesso colore usato per la linea. Volendo disegnare un rettangolo vuoto va usato B invece di BF. La grafica ad alta risoluzione per il gioco è completa. Il resto del programma riguarda il movimento dell'alieno e del missile e l'attivazione degli schermi.

## MUOVERE L'ALIENO

Al programma vanno aggiunte tre subroutine. La prima riguarda il movimento dell'alieno: trascriviamola, ma aspettiamo a dare il RUN, perché per adesso non ha alcun effetto.

```
1000 LETLX=AX:LETLY=AY
1010 IF RND(10)=1 THEN LETAX=RND
     (248)-1:LETAY=RND(178)+5
1020 LETAX=AX+RND(15)-8:LETAY=
     AY+RND(15)-8
1030 IF AX>103 AND AX<144 AND AY
     >71 AND AY<112 THEN LETAX
     =LX:LETAY=LY
1040 IFAX<0 THEN LETAX=-AX
1050 IFAX>248 THEN LETAX=497-AX
1060 IFAY<6 THEN LETAY=12-AY
1070 IFAY>184 THEN LETAY=369-AY
1080 PUT(LX,LY)-(LX+7,LY+7),BL,PSET
1090 PUT(AX,AY)-(AX+7,AY+7),AL,PSET
1100 RETURN
```

L'alieno si controlla in modo simile al movimento del missile e della base mobile esposto alle pagine 54-59. La linea 1ØØØ pone le coordinate dell'ultima posizione uguali a quella della posizione corrente prima che l'alieno si muova.

Per permettere all'alieno di spostarsi improvvisamente sullo schermo, la linea 1Ø1Ø sceglie un numero a caso. Se il numero a caso è 1, allora l'alieno balza in una nuova posizione sullo schermo. Se il numero non è 1, viene scelta una nuova posizione per l'alieno tra −7 e +7 pixel nella direzione x (da sinistra a destra) e lo stesso arco di distanza nella direzione y (in alto e in basso): vedere la linea 1Ø2Ø.

La linea 1Ø3Ø evita che l'alieno si scontri con la stazione, mentre le linee da 1Ø4Ø a 1Ø7Ø evitano che esso fuoriesca dallo schermo.

La linea 1Ø8Ø fa scomparire l'alieno disponendo una serie di spazi vuoti sopra la sua ultima posizione e la linea 1Ø9Ø lo fa ricomparire in una posizione nuova.

La linea 11ØØ riporta il programma alla linea 16Ø, che non è stata ancora digitata.

## LANCIO DEI MISSILI

La seconda subroutine decide se lanciare un missile, poi ne assegna la posizione sullo schermo e, infine, controlla quale scudo protettivo può fermare il missile.

```
2000 IFRND(7)<>1 THENRETURN
2010 V=RND(8)+5:DX=FNZ(127-AX):DY
     =FNZ(95-AY)
2020 IF DX<=0 AND DY>=0 THEN
     LETMA=1:GOTO 2050
2030 IF DX<=0 AND DY<=0 THEN
     LETMA=2:GOTO 2050
2040 IF DX>=0 AND DY<=0 THEN
     LETMA=3 ELSE LETMA=4
2050 LETMX=AX:LETMY=AY
2060 PUT(MX,MY)-(MX+4,MY+4),BO,OR
2070 LETAF=1:RETURN
```

La linea 2ØØØ decide se lanciare un missile. C'è una possibilità su sei per il lancio, ma il programma non lo esegue se c'è già un missile sullo schermo. Se non deve essere lanciato alcun missile, la subroutine termina. La linea 2Ø1Ø stabilisce la velocità del missile: si interpreti pure V come velocità. V è contenuta in FNZ (definita, a sua volta, nella linea 3Ø) per poter calcolare la nuova posizione del missile. Le linee da 2Ø2Ø a 2Ø4Ø ricercano il quarto di schermo in cui si trova il missile e quale scudo serve per parare il missile: se si vuole, MA è l'angolazione del missile.

La linea 2Ø5Ø lancia il missile dalla posizione dell'alieno e la linea 2Ø6Ø lo posiziona sullo schermo prima che la linea 2Ø7Ø renda AF=1, avvertendo così il Dragon dell'avvenuto lancio. La terza e ultima subroutine va dalle linee 3ØØØ a 3Ø7Ø.

Ancora una volta, trasciviamo ma non eseguiamo le seguenti linee:

```
3000 PUT(MX,MY)-(MX+4,MY+4),BL,
     PSET
3010 LETMX=MX+DX:LETMY=MY+DY
3020 IFMX>110ANDMX<140ANDMY>79
     AND MY<108 THEN GOTO 3050
3030 PUT(MX,MY)-(MX+4,MY+4),BO,OR
3040 RETURN
3050 IF SH(MA)=0 THEN GOTO 3070
3060 LETAF=0:RETURN
3070 CLS:PRINT@256"CRASH...
     GLI SCUDI ERANO ABBASSATI!"
```

La linea 3ØØØ cancella il missile. La sua nuova posizione è calcolata dalla linea 3Ø1Ø. La 3Ø2Ø controlla se il missile ha raggiunto gli scudi. Se sì, il programma salta alla linea 3Ø5Ø che verifica se lo scudo di destra è sollevato, altrimenti il programma finisce con il messaggio CRASH ... GLI SCUDI ERANO ABBASSATI!

Se invece il missile non ha raggiunto gli scudi protettivi, la linea 3Ø5Ø lo dispone sullo schermo nella nuova posizione. La linea 3Ø4Ø chiude la subroutine.

Ecco di seguito il completamento del programma:

```
150 SCREEN 1,1
160 IFAF=0 THEN GOSUB2000
    ELSEGOSUB3000
170 GOSUB1000
180 FOR J=1 TO 4
190 LETPE=PEEK(338+J):IF PW<
    25 THEN LETPE=255
200 IF 255-PE <> SH(J) THEN LETSH(J)
    =1-SH(J):CIRCLE(127,95),16,5*SH(J),1,
    (J+2)/4,(J+3)/4
```

```
210 IFSH(J)=1 THEN LETPW=PW-2
220 NEXT
230 LINE(PW,1)-(PW+2,5),PRESET,BF
240 GOTO160
```

Sul Tandy, si sostituisca la linea 20 con:

```
200 IF(255-PE)/16<>SH(J)THEN LET
    SHH(J)=1-SH(J):CIRCLE(127,95),
    16,5*SH(J),1(J+2)/4,(J+3)/4
```

Prima di lanciare il programma si tolgano le linee 15 e 145. Con l'eliminazione della linea 145 non si assiste alla creazione delle immagini: c'è una breve pausa, dopo il RUN, prima della loro completa comparsa.

Adesso è la linea 150 che mantiene abilitato lo schermo. La 160 verifica se è stato lanciato un missile: se AF=0, tale risposta è negativa e il programma salta alla subroutine per il lancio di un missile (quella che inizia alla linea 2000), altrimenti alla subroutine che muove il missile (quella che inizia alla linea 3000). Adesso l'alieno viene spostato. La linea 170 fa saltare il programma alla subroutine che inizia alla linea 1000. La sezione di programma dalla linea 180 alle 220 riguarda l'attivazione degli scudi. La linea 190 controlla qual'è il tasto premuto: se il tasto è un numero da 1 a 4, la linea 2000 disegna lo scudo di protezione e, quando questo è attivo, la linea 210 sottrae energia.

La linea 230 disegna un rettangolo nero dove viene visualizzato il consumo di energia, per rendere l'idea della riserva che diminuisce in corrispondenza con PW.

Infine, con la linea 240, il procedimento ricomincia.

Sugli apparecchi Acorn, il gioco della stazione spaziale si compone di due parti principali. La prima visualizza la grafica sullo schermo e si occupa, tra l'altro, di assegnare il valore iniziale alle variabili e di definire i caratteri UDG. La seconda, invece, concerne l'azione del gioco, come il movimento dell'alieno e il lancio del missile. Questa suddivisione facilita la comprensione del meccanismo di gioco.

## LA VISUALIZZAZIONE SULLO SCHERMO

Per capire la preparazione del gioco, si scrivano e si eseguano queste linee:

```
10 MODE1
20 LETX=RND(1100)+32:LETY=RND(950)
   +32
30 LETARMED=1:LETSH=1:LETF=1280
40 DIM C(4)
50 *FX11,1
60 VDU23,224,60,126,219,219,126,60,90,153
65 VDU23,225,32,78,89,124,62,154,114,4
70 MOVE0,1000:MOVE1280,1000:PLOT85,0,
   1024:PLOT85,1280,1024
80 PRINT "CARBURANTE":VDU5
90 MOVE560,512:DRAW640,592:DRAW720,
   512:DRAW640,432:DRAW560,512
100 MOVE640,592:DRAW640,432:MOVE560,
   512:DRAW720,512
110 MOVE592,552:PRINT "4□1":MOVE592,
   504:PRINT "3□2"
```

Le prime linee assegnano valori alle variabili. X e Y sono la posizione di partenza dell'alieno, ARMED=1 significa che l'alieno inizia con un missile (ARMED=0 nessun missile), SH=1 significa che gli scudi sono in funzione (0 significa che non lo sono) e F è il livello dell'energia disponibile. DIM C(4) dimensiona la matrice per il colore dei quattro schermi e *FX11,1 sveltisce la ripetizione automatica dei tasti.

Le linee 60 e 65 definiscono i caratteri UDG per l'alieno e per il missile, che verranno utilizzati in un'altra parte del programma.

Le linee successive si occupano della visualizzazione. La 70 e l'80 disegnano il quadrante dell'energia e le linee 90-110 disegnano la stazione spaziale, numerandone i settori da 1 a 4. La VDU 5, alla linea 80, fa sì che il testo sia visualizzato dal cursore grafico e questo semplifica la comparsa dei numeri dei settori nella giusta posizione.

Si noti come la linea 110 sposti il cursore prima di visualizzare i numeri.

## CREARE L'AZIONE

Il programma vero e proprio è di 7 linee! Eccole:

```
120 REPEAT
130 PROCALIENO
140 PROCSCUDO
150 PROCSPARI
160 PROCCARB
170 UNTIL FALSE
180 END
```

Ovviamente, serviranno altre linee per definire ogni procedura, ma la struttura del gioco resta semplicissima.

Le linee 100 e 150 creano un ciclo intorno alla lista delle procedure (UNTIL FALSE significa 'continua per sempre'). PROCALIENO sposta l'alieno, PROCSCUDO attiva gli scudi protettivi attorno alla base spaziale, PROCSPARI lancia i missili e verifica se è sollevato lo scudo giusto e PROCCARB misura la quantità di energia a disposizione.

Avviando il programma, adesso si ottiene un errore "No such FN/PROC", dal momento che nessuna procedura è stata definita: questo è il prossimo passo.

## MUOVERE L'ALIENO

Ecco le linee per muovere l'alieno:

```
190 DEF PROCALIENO
200 GCOL0,0:MOVEX,Y:VDU224
210 IF RND(200)=123 THEN X=640:Y=
    512:GOTO 280
220 LETDX=RND(40)-20:LETDY=RND(40)
    -20
230 IFX>1200 THEN DX=-ABS(DX)
240 IF X<10 THEN DX=ABS(DX)
250 IF Y<50 THEN DY=ABS(DY)
260 IF Y>950 THEN DY=-ABS(DY)
270 LETX=X+DX:LET Y=Y+DY
280 IF X>500 AND X<750 AND Y>400
```

lore Ø) a ogni scudo (uno scudo attivato è bianco). La linea 33Ø controlla se gli scudi sono in funzione: se SH=Ø, l'energia è esaurita e non si possono usare gli scudi.

Le linee successive, in base al tasto premuto, cambiano il colore dello scudo corrispondente (3=bianco). La linea 41Ø diminuisce l'energia di 2 unità quando viene usato uno scudo: economia nel loro uso, dunque!

Infine, la linea 42Ø disegna tutti e quattro gli scudi, anche se si vede soltanto quello bianco: gli altri tre sono neri e quindi invisibili. Questo è un modo semplice per visualizzare lo scudo giusto senza troppe linee di programma supplementari e complesse. Se però SH alla linea 33Ø fosse Ø, allora tutti e quattro gli scudi sarebbero neri.

```
    ANDY<650 THEN X=RND(125Ø): Y
    =RND (99Ø)+3Ø:GOTO28Ø
29Ø GCOLØ,2:MOVEX,Y:VDU224
3ØØ ENDPROC
```

Si provi ora a eseguire un RUN. Funzionerà aggiungendo una linea provvisoria:

```
135 GOTO17Ø
```

Si ricordi di cancellarla, dopo aver constato che la procedure funziona, scrivendo 135 seguito da RETURN.

L'alieno si sposta casualmente, per lo più a piccoli passi, ma ogni tanto balzando in un punto diverso dello schermo.

La linea 22Ø calcola i piccoli incrementi casuali DX e DY e la 21Ø controlla i balzi lunghi attraverso lo schermo. Il balzo avviene soltanto se RND (2ØØ) è uguale a 123: il numero 123 è un numero "di comodo" e potrebbe essere sostituito con qualsiasi numero tra 1 e 2ØØ. Ciò significa che c'è una possibilità su duecento che il balzo avvenga. Se si vuole far balzare l'alieno più spesso, i numeri vanno sostituiti con altri più bassi, per esempio, RND(1ØØ)=5Ø.

Le linee da 23Ø a 26Ø verificano se l'alieno si avvicina al margine dello schermo e, se ciò accade, lo riportano al centro sottraendo oppure aggiungendo le quantità DX e DY.

La linea 27Ø aggiunge un piccolo incremento e la 28Ø riporta l'alieno in una nuova posizione, se si avvicina troppo alla stazione. La linea 29Ø seleziona il colore giallo per la grafica e disegna l'alieno. VDU224 corrisponde a PRINT CHR$224 che è il numero di codice dell'UDG per l'alieno.

Manca ancora da spiegare la linea 2ØØ: il suo compito è semplicemente quello di cancellare la posizione precedentemente occupata dall'alieno.

## ATTIVARE GLI SCUDI

La successiva procedura di cui dobbiamo occuparci è PROCSCUDO:

```
31Ø DEF PROCSCUDO
32Ø LETC(1)=Ø:LETC(2)=Ø:LETC(3)=
    Ø:LETC(4)=Ø
33Ø IF SH=Ø THEN GOTO 42Ø
34Ø LETA$=INKEY$(1)
35Ø *FX15,1
36Ø IF A$=" "THEN GOTO 42Ø
37Ø IF A$="1" THEN C(1)=3
38Ø IF A$="2" THEN C(2)=3
39Ø IF A$="3" THEN C(3)=3
4ØØ IF A$="4" THEN C(4)=3
41Ø LETF=F-2
42Ø GCOL Ø,C(1):MOVE64Ø,6Ø8:DRAW736,
    512:GCOL Ø,C(2):DRAW64Ø,416:GCOL Ø,C
    (3):DRAW544,512:GCOL Ø,C(4):DRAW64Ø,
    6Ø8
43Ø ENDPROC
```

Ciò visualizza lo scudo che protegge la stazione spaziale dai missili. Può venir protetto soltanto un settore alla volta e gli scudi si attivano premendo uno dei tasti da 1 a 4.

La linea 32Ø assegna il colore nero (va-

## IL LANCIO DEI MISSILI

PROCSPARI dà all'alieno una possibilità su 3Ø di lanciare un missile contro la stazione spaziale e verifica se questo oltrepassa le barriere. Digitiamo queste linee:

```
44Ø DEF PROCSPARI
45Ø IF ARMED=Ø THEN GOTO 54Ø
46Ø IF RND(3Ø)<>13 THEN GOTO 59Ø
47Ø LETARMED=Ø:LETFX=(624-X)/15:
    LETFY=(528-Y)/15
48Ø LETG=X+FX:LETH=Y+FY
49Ø GCOL3,2:MOVEG,H:VDU225
5ØØ IF G>624 AND H>528 THEN S=1
51Ø IF G>624 AND H<528 THEN S=2
52Ø IF G<624 AND H<528 THEN S=3
53Ø IF G<624 AND H>528 THEN S=4
54Ø GCOL3,2:MOVEG,H:VDU225
55Ø LETG=G+FX:LETH=H+FY
56Ø IF NOT(G>584 AND G<684 AND H
    >48Ø AND H<58Ø) THEN MOVEG,H:
    VDU225:ENDPROC
57Ø LETARMED=1
58Ø IF C(S)=Ø THEN PROCFINE
59Ø ENDPROC
```

La linea 45Ø verifica se l'alieno è armato. Se ARMED è uguale a Ø, è già stato sgan-

ciato un missile e il programma salta ad una parte successiva. La linea 460 salta alla fine della procedura, a meno che RND(30) = 13. Il 13 è un altro numero di comodo come il 123 alla linea 210. Dà all'alieno una possibilità su 30 di lanciare un missile.

Posto che le condizioni alle linee 450 e 460 non siano vere, allora il computer raggiunge la linea 470, che lancia immediatamente un missile e poi calcola la distanza in FX e FY: questi sono regolati in modo che il missile punti sempre contro la stazione. La linea 480 aggiunge questo risultato alla posizione dell'alieno per dare la posizione del missile: G e H sono le coordinate del missile. La linea 490 disegna un missile giallo.

Le quattro linee successive calcolano in quale settore si trova il missile e assegnano la variabile S al numero del settore.

La linea 540 cancella la vecchia posizione del missile, la 550 lo avanza di un passo e la 560 lo visualizza nella nuova posizione se *non* è vicino alla stazione.

Il programma arriva alla linea 570 soltanto quando il missile raggiunge la base. L'alieno torna subito ad essere armato (ARMED = 1) e viene controllato il colore dello scudo protettivo. Se questo è abbassato, ossia di colore nero, il gioco finisce richiamando la PROCFINE. Altrimenti il gioco procede normalmente.

## LA RISERVA DI ENERGIA

La PROCCARB è la seguente:

```
600 DEF PROCCARB
610 LETF=F-.75
620 IF SH=0 THEN GOTO 650
630 GCOL0,0:MOVE F+10,1000:MOVE F+
    10,1024:PLOT 85,F,1000:PLOT 85,F,1024
640 IF F<131 THEN SH=0
650 ENDPROC
```

Questa procedura fa diminuire la riserva di carburante e la linea 630 cancella parzialmente il quadrante per segnalare il consumo. Si ricordi che l'energia cala più rapidamente con gli scudi attivati: si riconfronti la linea 410.

Se il carburante scende al di sotto di 131, la linea 640 abbassa gli scudi, riportando SH a 0.

## IL FINALE DEL GIOCO

Se un missile attraversa le barriere della stazione abbiamo perso la partita. Di ciò si occupa PROCFINE:

```
660 DEF PROCFINE
665 FOR D=1 TO 2000: NEXT
670 VDU4:CLS:PRINT''"CRASH! GLI SCUDI
    ERANO ABBASSATI"
680 *FX12,0
685 FOR D=1 TO 2000: NEXT
690 *FX15,1
700 END
```

Il suo compito è di notificarci che siamo spacciati, ridisponendo le varie funzioni precedentemente attivate. VDU 4 disattiva il comando VDU 5. *FX12,0 riporta la ripetizione automatica dei tasti alla norma e *FX15,1 svuota il buffer della tastiera pieno degli 1, 2, 3 e 4 premuti durante il gioco. In breve, serve a "rimettere tutto a posto", prima di concludere il gioco.

La versione Spectrum adopera svariate nuove caratteristiche, non ancora spiegate, che richiedono perciò un po' di attenzione e di intuizione.

Come sempre, per controllare il funzionamento di ogni parte, si introdurranno le linee un poco alla volta: queste definiscono l'alieno e, dopo il RUN, lo visualizzano sullo schermo:

```
10 BORDER 0: PAPER 0: INK 6:
   BRIGHT 1: CLS
20 FOR n=USR "a" TO USR "b"+7: READ
   a: POKE n,a: NEXT n
200 LET ax=INT (RND*32)
210 LET ay=INT (RND*21)+1
220 IF ax>11 AND ax<21 AND ay>6
    AND ay<16 THEN GOTO 200
490 PRINT INK 4;AT ay,ax;CHR$ 144
800 DATA 60,126,219,219,126,60,90,153,0,0,
    24,60,60,24,0,0
```

Le linee 20 e 800 definiscono l'alieno e il suo missile (non ancora visibile). La tecnica usata è quella ampiamente esposta in Codice macchina 2. (For n = USR "a" TO USR "b" + 7 ... POKE n, a corrisponde a FOR n = 0 to 15 ... POKE USR "a" + n,a.)

Le linee 220 e 210 assegnano all'alieno una posizione a caso sullo schermo e la 490 lo visualizza. (PRINT CHR$ 144 in questa linea corrisponde alla PRINT ⟨"a" grafica⟩ della lezione precedente.)

A questo livello, la linea 220 sembra strana, ma serve, come si vedrà in seguito, a evitare che l'alieno sbuchi nel bel mezzo della stazione spaziale.

Si può temporaneamente omettere la linea 10, dal momento che un listato di programma giallo su schermo nero è scomodo da leggere. In tal caso, però, dobbiamo ricordarci di inserirla più tardi o la linea 640 (spiegata sotto) non funzionerà.

## LA STAZIONE SPAZIALE

Con queste poche linee si disegna la stazione:

```
110 PRINT AT 10,15;"4□1";AT 12,15;"3□2"
120 PLOT 132,107: DRAW 25,-25: DRAW
    -25,-25: DRAW -25,25: DRAW 25,25
130 PLOT 107,82: DRAW 50,0: PLOT 132,57:
    DRAW 0,50
```

Per adesso è una stazione un po' rudimentale. Se si vuole disegnarne una vera e propria, servono soltanto due aggiunte:

- Una linea simile alla 20, ma che inizi con USR "c" e che continui per un numero di lettere dell'alfabeto corrispondente alla misura richiesta per la stazione.
- Un lungo, lungo gruppo di frasi DATA in una o più linee alla fine del programma.

## VISUALIZZIAMO IL MISSILE

Dobbiamo riprodurre il missile dell'alieno e tracciarne il percorso verso la base spaziale:

```
150 LET mf=0
300 IF mf=1 THEN GOTO 400
310 IF RND<.9 THEN GOTO 420
320 LET mf=1: LET my=ay: LET mx=ax:
    LET fy=11-my: LET fx=16-mx
330 LET b=1: IF ABS fy>ABS fx THEN
    LET b=2
340 IF b=1 THEN LET sx=SGN fx: LET sy
    =SGN fy*ABS (fy/fx)
350 IF b=2 THEN LET sy=SGN fy: LET sx
    =SGN fx*ABS (fx/fy)
400 PRINT AT my,mx;"□": LET my=my
    +sy: LET mx=mx+sx: PRINT INK 5;AT
    my,mx;CHR$ 145: IF my>10 AND my
    <12 AND mx>15 AND mx<17 THEN
    GOTO 700
620 IF RND>9 THEN PRINT AT ay,ax;"□":
    GOTO 200
630 IF mf=0 THEN GOTO 300
650 GOTO 300
700 CLS: PRINT FLASH 1; PAPER 2;AT
    10,1;"CRASH!□Gli scudi erano
    abbassati□"
```

Tutta questa sezione, come si può vedere dalla linea 650, è un ciclo che il computer ripete più volte quando appare l'alieno.

La linea 150 prepara lo scenario: non c'è ancora nessun missile in arrivo contro la stazione.

La linea 310 decide se l'alieno lancerà un missile durante un particolare ciclo del programma (c'è una possibilità su nove che lo faccia). Se capita il lancio, la linea 320 assegna la sua posizione di partenza (my, mx) nella posizione dell'alieno (ax, ay). La parte centrale della linea 400 è la trovata più importante: dai numeri corrispondenti al centro della stazione vengono sottratti quelli che rappresentano la posizione attuale dell'alieno, in modo che il missile punti alla stazione spaziale.

Poiché alcuni dei numeri possono essere negativi (nel percorso dalla sinistra verso il basso) e altri positivi (da destra verso l'alto), questa parte del programma può risultare piuttosto oscura, non conoscendo ancora le funzioni ABS e SGN, descritte in una lezione successiva. Ecco dunque qualche cenno.

La seconda metà della linea 320 deduce la posizione attuale del missile (mx, my) dal punto centrale della stazione (posizione sullo schermo 11, 16) e chiama le coordinate risultanti fy e fx.

Le linee da 330 a 350, con ABS e SGN, aggiungono fattori di 'correzione di rotta' (sy e sx) a fy e fx. La linea 400 cancella il missile dalla sua vecchia posizione, poi aggiunge a questa i numeri sx e sy, per visualizzarlo in una posizione leggermente avanzata.

## IL MOVIMENTO DELL'ALIENO

Avendo sparato il suo missile, l'alieno deve adesso spostarsi. Si aggiungano queste linee:

```
420 LET xx=ax: LET yy=ay: LET m=INT
    (RND*4)
430 IF m=0 AND ax<31 THEN LET xx
    =ax+1
440 IF m=1 AND ax>0 THEN LET xx=ax
    -1
450 IF m=2 AND ay<21 THEN LET yy
    =ay+1
460 IF m=3 AND ay>1 THEN LET yy=ay
    -1
470 IF xx>11 AND xx<21 AND yy>6
    AND yy<16 THEN GOTO 490
480 PRINT AT ay,ax;"□": LET ax=xx: LET
    ay=yy
```

Prima di tutto, lo Spectrum decide in quale direzione muovere l'alieno.

Ciò si ottiene con un numero casuale alla linea 420. Il compito delle linee da 430 a 460 è quello, al solito, di generare movimento. La linea 470 allontana l'alieno dalla stazione. La linea 400 (digitata in precedenza) devia l'esecuzione alla linea 700, se il missile colpisce in centro della stazione. Se non si afferra perché questa linea ricorra a >10 e <12 invece di un più semplice =11 e a >15 e <17 invece di =16, si rifletta sul fatto che i numeri impiegati per individuare la posizione dell'alieno sono frazioni decimali: le probabilità che capiti effettivamente un 11 o un 16 sono, quindi, remote.

Finalmente, circa al decimo giro, la linea 620 cancella l'alieno nella sua posizione finale di questo ciclo e riparte dalla linea 200.

## COSTRUIRE GLI SCUDI

Queste linee costruiscono gli scudi protettivi per respingere i missili:

```
140 PLOT INVERSE 1;132,122
500 DIM a(4)
510 LET a$=INKEY$: IF a$="" THEN
    GOTO 600
520 IF a$="1" THEN LET a(1)=1
530 IF a$="2" THEN LET a(2)=1
540 IF a$="3" THEN LET a(3)=1
550 IF a$="4" THEN LET a(4)=1
600 DRAW INK a(1)*4, INVERSE 1-a(1), 40,
    -40: DRAW INK a(2)*4, INVERSE 1
    -a(2),-40,-40: DRAW INK a(3)*4,
    INVERSE 1-a(3),-40,40: DRAW INK
    a(4)*4, INVERSE 1-a(4),40,40
640 IF ATTR (my,mx)=68 THEN PRINT AT
    my,mx;"□": LET mf=0
```

Anche in queste linee, a prima vista c'è qualcosa di strano: *quattro* scudi con soltanto *una* posizione per il PLOT che disegna le loro linee? Il programma in effetti usa lo stesso colore per l'oggetto e per lo sfondo nel disegnare un rombo. Soltanto premendo uno dei tasti numerati, una sezione del rombo cambia colore e compare sullo schermo.

Frattanto la linea 640 usa ATR 68, il numero per il colore degli scudi, per respingere il missile, cancellandolo se esso colpisce gli scudi.

## RIFINITURE

Le linee restanti si seguono con facilità: esse predispongono il livello di energia a 1000 e lo fanno diminuire finché, a metà, della linea 510 (modificata), gli scudi restano disattivati. Ricordiamoci di riscrivere la linea 10:

```
100 PRINT PAPER 2; INK 6;AT
    0,0;"□CARB□"
160 LET fu=100
510 LET a$=INKEY$: IF a$="" OR fu=0
    THEN GOTO 600
560 LET fu=fu-1
610 PRINT PAPER 3; INK 7;AT
    0,6;"□";fu;"□"
```

La versione del gioco per il Commodore 64 sfrutta gli sprite, le informazioni per i quali sono contenute in numerose frasi DATA all'inizio del programma, come spiegato a pagina 15.

```
10 POKE 56,100:POKE 55,0:POKE
   52,100:POKE 51,0:CLR
```

```
20 DATA 0,254,0,3,57,128,7,255,192,0,16,0,
   16,56,16,56,84,56,124,146,124,131,255
30 DATA130,144,56,18,184,16,58,144,16,18,
   131,255,130,254,84,254,252,55,126,0,56
40 DATA0,0,40,0,0,56,0,1,199,0,6,16,192,1,
   199,0,0,124,0
50 DATA0,0,0,127,255,254,64,0,2,64,0,2,64,
   16,2,64,48,2,64,16,2,64,16,2,80,56
60 DATA114,84,0,18,94,0,114,68,0,66,68,56,
   114,64,8,2,64,56,2,64,8,2,64
70 DATA56,2,64,0,2,64,0,2,127,255,254,0,0,0
80 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
   0,0,8,0,0,42,0,0,20,0,0,42,0,0
90 DATA42,0,0,73,0,0,8,0,0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,0
100 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,3,3,0,4,132,128,255
110 DATA255,255,72,72,68,48,48,56,0,0,0,0,
   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
120 DATA0,0,16,0,0,24,0,0,20,0,0,20,0,0,20,0,
   0,24,0,0,48,0,0,80,0,0,80,0,0,48,0
130 DATA0,24,0,0,20,0,0,20,0,0,24,0,0,48,0,0,
   80,0,0,80,0,0,80,0,0,48,0,0,16
140 DATA0,0,16,0
210 FOR ZZ=0 TO 4:POKE 2040+ZZ, 200
   +ZZ
220 FOR Z=1TO63:READ X:POKE 12799+
   (ZZ*64)+Z,X:NEXT Z,ZZ
230 CLR:V=53248:FU=100:POKE 650,255:
   POKE 53280,0:POKE 53281,0:PRINT "♡"
240 POKE V+2,145:POKE V+3,120:POKE V
   +23,250:POKE V+29,250:POKE V+
   30,240
250 XX=31+INT(RND(1)*210):YY=60:DX
   =1:DY=1:IF RND(1)>.50 THEN YY=
   180:DY=-DY
260 PRINT "▣"TAB(14)"πCARB:□□□□
   ████████" FU
270 IFRND(1)>.90THENXX=31+INT(RND
   (1)*210):YY=60:DX=1:DY=1
320 XX=XX+DX: IF XX=<3 OR XX=
   >245 THEN DX=-DX
330 YY=YY+DY:IF YY=<50 OR YY=
   >190 THEN DY=-DY
340 POKE V,XX: POKE V+1,YY:IF F=
   0 THEN F=1:FX=XX:FY=YY
350 IF F=1 THEN GOSUB 410
360 GET A$:S$=" ":SH=0
370 IF A$="4" OR A$="2" THEN SH=1:
   POKE 2043,204:S$=A$
380 IF A$="1" OR A$="3" THEN SH=1:
   POKE 2043,203:S$=A$
390 IF SH=1 THEN 470
400 POKE V+21,247:GOTO 260
410 IF FX>153 THEN FX=FX-5
420 IF FX<153 THEN FX=FX+5
430 IF FY<135 THEN FY=FY+5
440 IF FY>135 THEN FY=FY-5
450 POKE V+4,FX:POKE V+5,FY:IF PEEK(V
   +30)=246 THEN 550
460 RETURN
470 IF S$="4" THENL1=118:L2=120
480 IF S$="2" THENL1=175:L2=120
490 IF S$="1" THENL1=145:L2=95
500 If S$="3" THENL1=145:L2=145
510 POKE V+6,L1:POKE V+7,L2:FU=FU
   -1:POKE V+21,255:IF FU<0THEN 540
520 IF PEEK(V+30)=252 THEN F=0:GOTO
   240
530 GOTO 260
540 PRINT TAB(4)"▙ ▤IL CARBURANTE È
   ESAURITO!":GOTO 560
550 PRINT TAB(11)"▙ ▤SEI STATO
   COLPITO!"
560 FOR ZZ=1TO10:FOR T=1TO100:NEXT:
   POKE V+21,247:FOR T=1TO100:NEXT:
   POKE V+21,0
570 NEXT:PRINT "♡"TAB(12)"▙▙▙▙
   ▙▙▙▙▙▙▙▙▒▟
   PREMERE LA BARRA SPAZIATRICE"
580 GET X$:IFX $<>"□" THEN 580
590 RUN 230
```

La prima linea del programma riserva spazio per lo sprite nella memoria del Commodore, affinché il programma in BASIC usato per muovere e operare sullo sprite non interferisca con il programma per lo sprite.

Le linee dalla 20 alla 140 contengono l'informazione per gli sprite della stazione spaziale, degli scudi, dei missili e dell'alieno.

Le linee 210 e 220 definiscono i *puntatori* degli sprite e depositano le relative DATA in memoria. Ciascun dei cinque sprite occupa 64 byte (3 volte 21, più uno): ciò spiega il significato di questi valori nelle due linee del programma.

Le due linee successive inizializzano il computer assegnando le variabili, le posizioni dei vari sprite, la ripetizione automatica ed i colori.

Alla linea 250 il programma assegna una posizione casuale allo sprite dell'alieno, che compare lungo una linea in alto o in basso a seconda del valore ottenuto dalla funzione RND.

Si comincia a osservare il programma in azione dalla linea 260: sullo schermo appaiono l'indicazione sul carburante e l'alieno. Questi lancia un missile e poi si sposta.

Le linee 360-400 attivano gli scudi protettivi in corrispondenza della pressione dei tasti, 1, 2, 3 e 4.

Uno scudo sollevato soddisfa la condizione alla linea 390 e il programma salta alla routine alle linee 470-530, la quale controlla la posizione dello sprite sullo schermo.

Il missile punta sempre all'obiettivo, grazie alla routine compresa tra la linea 410 e la 460.

Se lo sprite del missile riesce a raggiungere il centro della stazione, ossia se alla linea 520 non si constata una collisione fra sprite, la condizione alla linea 450 risulta soddisfatta.

La routine di fine gico, dalla linea 540 in poi, visualizza un messaggio, poi lo schermo lampeggia e infine viene proposta un'altra partita.

Ogni volta che viene attivato lo schermo, il contatore dell'energia, la variabile FU, è decrementato.

I caratteri di controllo del cursore, contenuti nella linea 260, che visualizza il livello d'energia, riportano indietro il cursore, cancellando così il valore del consumo precedente, prima di aggiungere il nuovo valore di FU. Qualora il nuovo valore di FU risultasse minore di 0, alla linea 510, l'esecuzione devierebbe verso la parte di programma che conclude il gioco.

# MATRICI: I MAGAZZINI DELL'INFORMAZIONE

**Le statistiche, di qualsiasi tipo esse siano, sono il cavallo di battaglia dei computer. Uno degli strumenti più efficaci per elaborarle è la matrice**

Le matrici sono il metodo con cui il programmatore manipola grandi quantità di informazioni: lunghe liste di membri di una società, con relativi indirizzi e sottoscrizioni, dati finanziari complessi ed anche liste di personaggi, armi e tesori per i giochi di avventura.

Volendo archiviare in una lista di una certa lunghezza ciascun elemento dell'informazione individuale, si dovrebbe ricorrere ad un uso massiccio di istruzioni LET, ottenendo un programma lungo da scrivere e da eseguire.

La peculiarità delle matrici sta nel fatto che esse contengono informazioni in una forma molto compatta: non occorrono variabili diverse per ogni elemento dell'informazione, ma una sola comune a tutte: per esempio, A. Per differenziare tra loro i singoli elementi, si usa semplicemente un numero (a volte una lettera) tra parentesi. Così il primo elemento si indica con A(1), il secondo con A(2) e così via.

Una matrice non fornisce soltanto un contenitore più compatto, ma permette anche di cambiare ogni elemento (un numero telefonico, per esempio) con la massima facilità.

## PREPARARE UNA MATRICE

Per utilizzare una matrice, è necessario comunicare al computer la sua lunghezza, per permettergli di riservare abbastanza spazio in memoria. Ciò viene fatto con l'istruzione DIM (da 'dimensione'), come in questa linea:

```
10 DIM A(3)
```

Sullo ZX81 si usi una A maiuscola.

```
10 DIM a(4)
```

Ciò avverte il computer che questa particolare matrice ha quattro elementi, o variabili. Sugli Acorn, Commodore, Dragon e Tandy, questi saranno numerati da A(Ø) ad A(3). Sui Sinclair, che non accettano l'elemento a(Ø), saranno invece numerati da a(1) ad a(4). Il numero degli elementi usati può essere grande quanto si desidera e non c'è obbligo di usare tutti gli elementi che si riservano. Anzi, è consuetudine riservare qualche elemento in più. Si ricordi, però, che, esagerando, non rimarrà posto per altre variabili, causando un errore del tipo "out of memory".

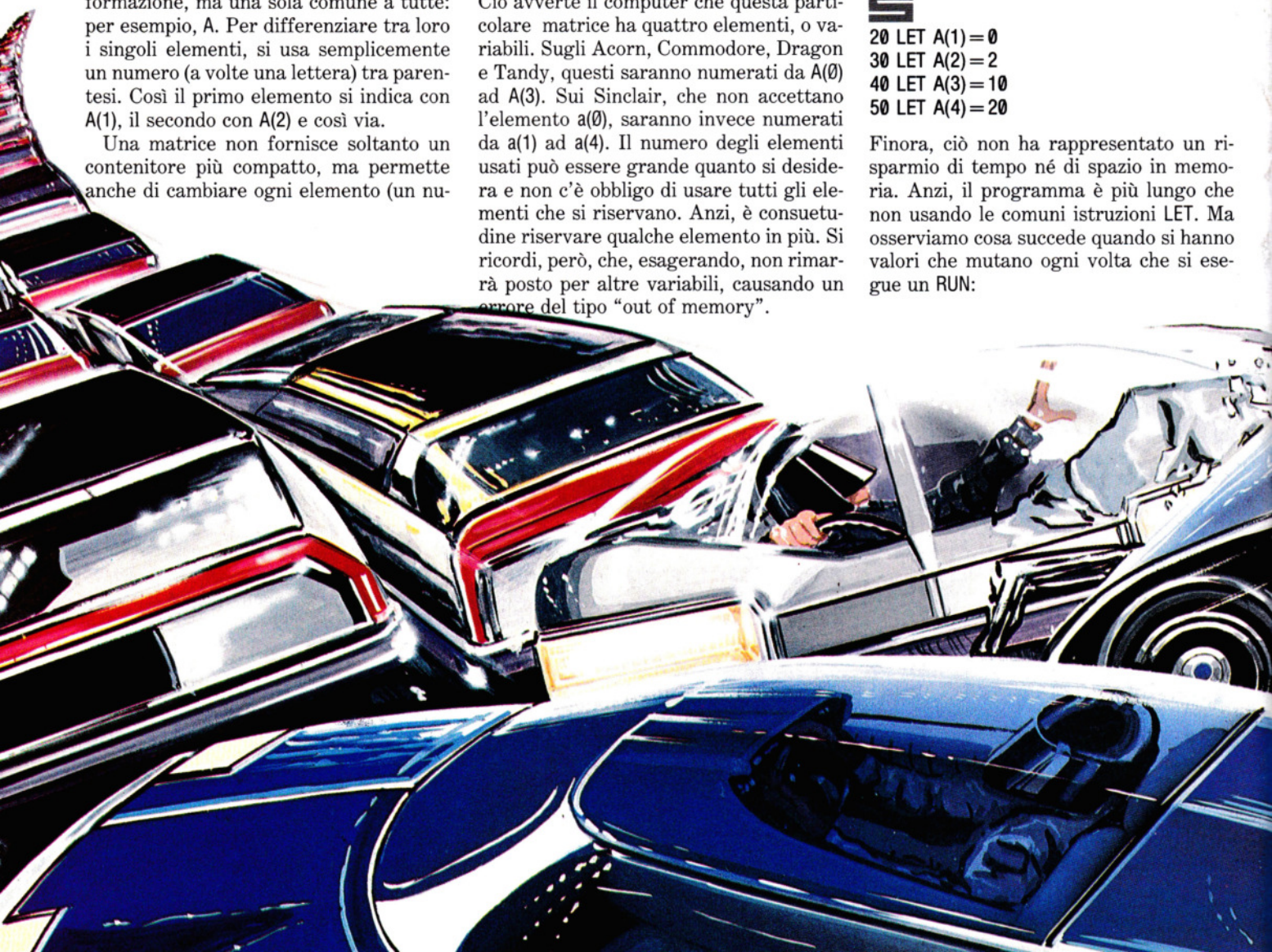
## ASSEGNARE I VALORI

Compito successivo è assegnare i valori a ogni elemento. Se, per esempio, le variabili rappresentassero le posizioni sullo schermo in cui visualizzare testi o grafica, la successiva linea del programma potrebbe essere:

```
20 LET A(Ø)=Ø: LET A(1)=2: LET A(2)
   =1Ø: LET A(3)=2Ø
```

```
20 LET a(1)=Ø: LET a(2)=2: LET a(3)=1Ø:
   LET a(4)=2Ø
```

```
20 LET A(1)=Ø
30 LET A(2)=2
40 LET A(3)=1Ø
50 LET A(4)=2Ø
```

Finora, ciò non ha rappresentato un risparmio di tempo né di spazio in memoria. Anzi, il programma è più lungo che non usando le comuni istruzioni LET. Ma osserviamo cosa succede quando si hanno valori che mutano ogni volta che si esegue un RUN:

```
10 DIM A(3)
20 PRINT "Quali sono i valori?"
30 INPUT A(0), A(1), A(2), A(3)
```

```
10 DIM a(4)
20 PRINT "Quali sono i valori?"
30 INPUT a(1),a(2),a(3),a(4)
```

```
10 DIM A(4)
20 PRINT "QUALI SONO I VALORI"
30 INPUT A(1)
40 INPUT A(2)
50 INPUT A(3)
60 INPUT A(4)
```

A ogni RUN, il computer chiede i valori da assegnare a ciascuna variabile e sarà sufficiente, in ciascun caso, immettere un numero (seguito naturalmente da ENTER o RETURN).

Qualora la quantità degli elementi sia molto grande, si potrà apprezzare il notevole risparmio di tempo. Supponiamo, per esempio, che si vogliano digitare cento numeri. Basterà questo semplice programma:

```
10 DIM A(99)
20 FOR N=0 TO 99
30 INPUT A(N)
40 NEXT N
```

Sullo ZX81 si scriva tutto in maiuscole.

```
10 DIM a(100)
20 FOR n=1 TO 100
30 INPUT a(n)
40 NEXT n
```

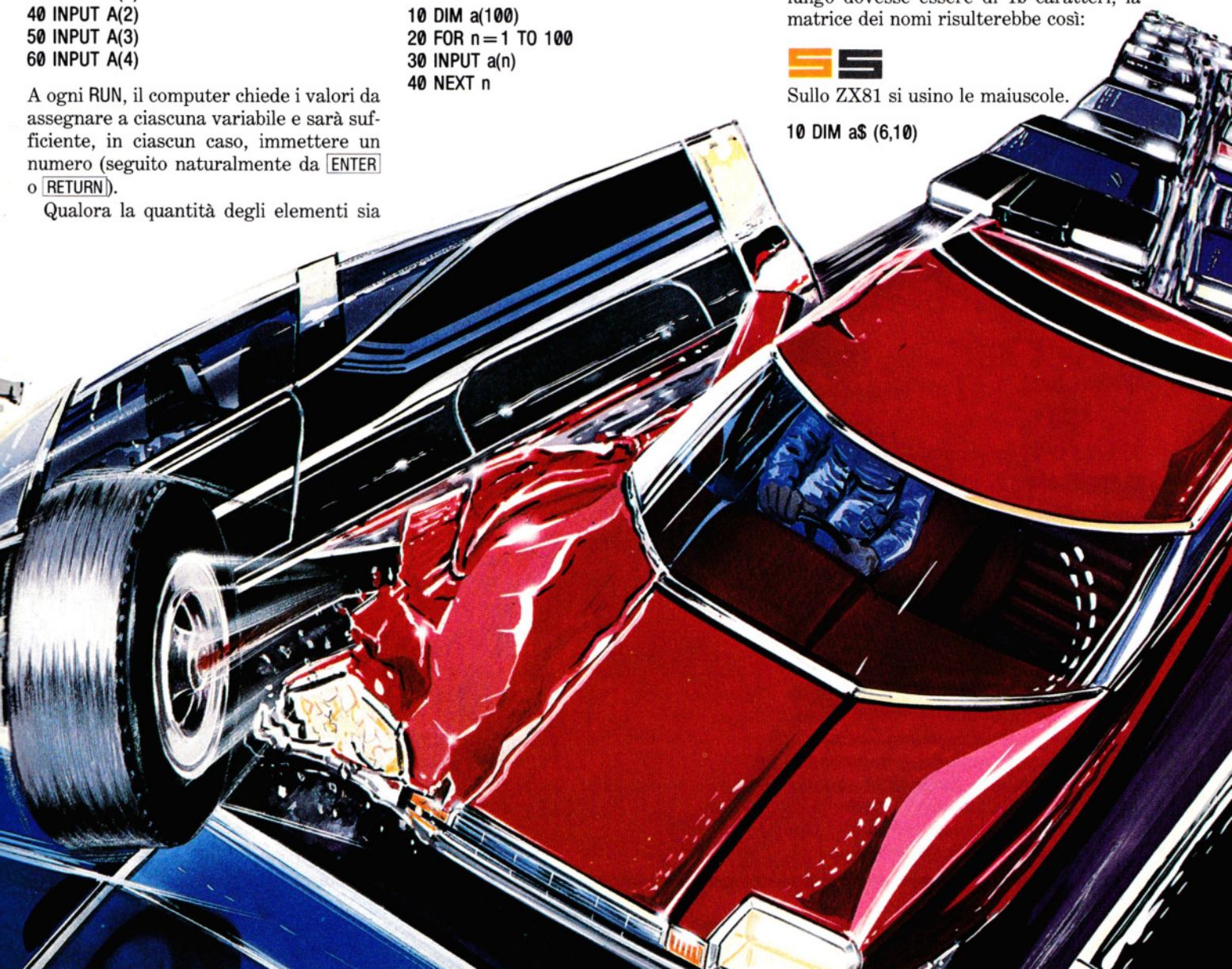
## TRATTAMENTO DEI NOMI

Il tipo di matrice finora descritto permette di trattare soltanto numeri. Se si vogliono elaborare sia nomi che numeri, occorrono due matrici: una per i nomi e l'altra per i numeri. Su tre dei nostri apparecchi, la matrice dei nomi risulterà così:

```
10 DIM A$(5)
```

Sullo Spectrum, è necessario specificare anche la lunghezza massima che il nome può raggiungere. Perciò, se il nome più lungo dovesse essere di 10 caratteri, la matrice dei nomi risulterebbe così:

Sullo ZX81 si usino le maiuscole.

```
10 DIM a$ (6,10)
```

In ogni caso, il computer ha riservato spazio in memoria per sei nomi o sei parole. Per preparare un ciclo di immissione, si scriva:

```
20 FOR N=0 TO 5
30 INPUT A$(N)
40 NEXT N
```

Sullo ZX81 si scriva tutto in maiuscolo:

```
20 FOR n=1 TO 6
30 INPUT a$(n)
40 NEXT n
```

Lanciamo il programma e immettiamo ogni nome o lettera, seguiti da ENTER o RETURN. Poi verifichiamolo aggiungendo la seguente linea:

```
35 PRINT A$(N)
```

Sullo ZX81 va scritta in maiuscole.

```
35 PRINT a$(n)
```

Anche se i nomi fossero lunghi, non è difficile digitarne sei e risulta abbastanza pratico digitarne anche un centinaio.

Supponiamo, ad esempio, di voler fare un'analisi su dei percorsi automobilistici e di voler registrare i nomi delle curve e il numero di incidenti capitati in ciascuna durante un anno. Si arriverebbe con facilità a immettere il nome di una ventina di curve e con altri esempi sarebbero necessarie anche più immissioni, ma il principio resta identico a quello del seguente programma, in cui sono soltanto sei (questo metodo ricorre all'impiego di frasi DATA per cui non è utilizzabile sullo ZX81 che non ha questa semplificazione):

```
10 DIM A$(5)
20 FOR N=0 TO 5
30 READ A$(N)
40 NEXT N
50 DATA FIVE MILE, WELL PASS, BROOK
   HILL, PETERS ROAD, CROSSWAYS,
   ROWLANDS
```

```
10 DIM a$(6,11)
20 FOR n=1 TO 6
30 READ a$(n)
40 NEXT n
50 DATA "FIVE MILE", "WELL PASS",
   "BROOK HILL", "PETERS ROAD",
   "CROSSWAYS", "ROWLANDS"
```

I nomi verranno letti dalle frasi DATA alla linea 50 e depositati nella matrice alla linea 10. Ogni volta che il programma è avviato, gli stessi nomi vengono immessi automaticamente. Se i nomi fossero 100, occorrerebbe modificare la linea 10 in DIM A$(99) e la 20 in FOR n=0 TO 99 (per gli Acorn, il Dragon, il Tandy e il Commodore) oppure DIM a$(100,11) e FOR n=1 TO 100 (per lo Spectrum) ed aggiungere gli altri 94 nomi nelle frasi DATA.

Si potrebbe poi preparare una matrice per includere il numero degli incidenti:

```
60 DIM A(5)
70 FOR N=0 TO 5
80 READ A(N)
90 NEXT N
100 DATA 0, 2, 5, 1, 3, 6
```

```
60 DIM a(6)
70 FOR n=1 TO 6
80 READ a(n)
90 NEXT n
100 DATA 0, 2, 5, 1, 3, 6
```

Volendo, si può ignorare l'elemento zero di una matrice sugli Acorn, Commodore, Dragon e Tandy e quindi nell'ultimo esempio scrivere DIM A$(6), numerando gli elementi da 1 a 6. Anche se ciò significherebbe che A$(0) è vuoto, resta più naturale contare da 1 invece che da 0.

## USO DELLE MATRICI

Adesso che il computer conosce il numero degli incidenti capitati ad ogni incrocio o curva, come elaborare queste informazioni? Una prima cosa utile potrebbe essere una stampa dell'elenco delle curve e degli incidenti, anche solo per controllare la correttezza di ciò che si è scritto.

Scriviamo le seguenti linee:

```
210 FOR N=0 TO 5
220 PRINT A$(N), A(N)
230 NEXT N
```

```
210 FOR n=1 TO 6
220 PRINT a$(n), a(n)
230 NEXT n
```

Le linee 210 e 230 creano un ciclo, nel quale la linea 220 visualizza i nomi ed i numeri delle matrici. Se si riscontrano errori, ora è il momento di correggerli.

Arrivati ad analizzare i risultati dell'esame, si vorrà sapere il numero totale degli incidenti o quale sia il luogo più sicuro. Le linee per calcolare il totale potrebbero essere queste:

```
300 PRINT"♥"
310 LET TL=0
320 FOR N=0 TO 5
330 LET TL=TL+A(N)
340 NEXT N
350 PRINT"NUMERO TOTALE DI
   INCIDENTI□";TL
```

```
300 CLS
310 LET totale=0
320 FOR N=0 TO 5
330 LET totale=totale+A(N)
340 NEXT N
350 PRINT "Numero totale di
   incidenti:□";totale
```

```
300 CLS
310 LET TL=0
320 FOR N=0 TO 5
330 LET TL=TL+A(N)
340 NEXT N
350 PRINT "NUMERO TOTALE DI
   INCIDENTI□";TL.
```

```
300 CLS
310 LET totale=0
320 FOR n=1 TO 6
330 LET totale=totale+a(n)
340 NEXT n
350 PRINT "Numero totale di
    incidenti:□";totale
```

## ANALISI DELL'INFORMAZIONE

Si immagini l'utilità di questo breve programma, se l'esame riguardasse 100 o 1000 curve o incroci, anziché solo sei.

Considerando, inoltre, che le informazioni di una matrice non solo possono essere memorizzate, ma anche facilmente analizzate, si comprenderà l'efficacia di un tale strumento, dai piccoli conti di casa, fino alla finanza internazionale. Ecco un esempio di questo genere di analisi:

```
400 FOR N=0 TO 5
410 IF A(N)>3 THEN PRINT A$(N),A(N)
420 NEXT N
```

```
400 FOR n=1 TO 6
410 IF a(n)>3 THEN PRINT a$(n),a(n)
420 NEXT n
```

Queste linee visualizzano un elenco di incroci nei quali sono avvenuti più di tre incidenti. Nel nostro esempio con sei incroci, questi sono BROOK HILL e ROWLANDS. Cambiando, alla linea 410, il 3 in 5 e riavviando il programma, viene visualizzato ROWLANDS. (Ogni numero maggiore di 6, che è il massimo, non porterebbe ad alcun risultato). L'informazione contenuta nelle matrici potrebbe anche essere un elenco di famiglie, insieme a statistiche sul numero dei figli, reddito o numero di auto possedute. Inoltre, i dati immessi si potrebbero riordinare secondo più criteri e si potrebbe poi chiedere al computer di trovare un nome singolo, anche ricordando solo la lettera iniziale. Per mostrare come ciò avverrebbe, si aggiungano queste linee al programma:

```
600 FOR N=0 TO 5
610 B$=A$(N)
620 IF LEFT$(B$,1)="P" THEN PRINT B$
630 NEXT N
```

```
600 FOR n=1 TO 6
620 IF a$(n,1)="P" THEN PRINT a$(n)
630 NEXT n
```

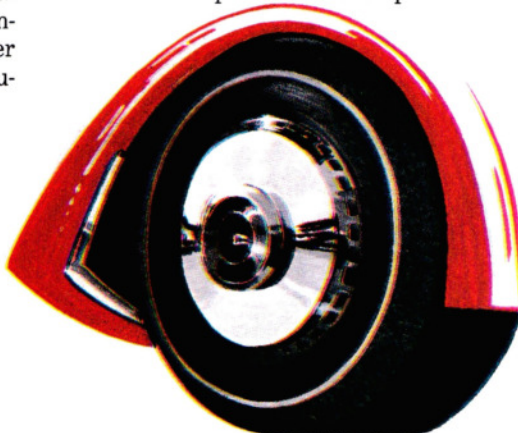
Le linee 600 e 630 preparano il ciclo per esaminare ciascun elemento della matrice. La linea 620 controlla se il primo carattere di ogni nome è una "P" visualizzando, in tal caso, l'intero nome.

Nel nostro esempio viene trovato PETERS ROAD, perché è il solo nome che inizia per 'P'. In un programma che elenca famiglie, la ricerca potrebbe riguardare tutti i Bianchi oppure i possessori di due macchine. Con matrici a più dimensioni, che sarà argomento della prossima lezione sulle matrici, si possono fare ricerche incrociate: per esempio, tutti i nomi che iniziano per 'A', di coloro che abitano al numero civico 21 e che possiedono un gatto siamese.

## MATRICI PER I GIOCHI

I giochi d'avventura sono un territorio letteralmente dominato dalle matrici. Invariabilmente, un'avvenutra prevede un numero di luoghi o scenari da attraversare. In ciascun luogo, appaiono sullo schermo le istruzioni per guidare il giocatore nelle varie fasi. Tutti i luoghi sono collegati tra loro ed è conveniente raccoglierli in una matrice stringa: A$ seguita da un numero, per esempio A$(9). Le direzioni scelte dal giocatore in ciascun luogo, ad esempio "nord", possono trovar posto in una seconda stringa; gli oggetti tipo "torce" e "chiavi" in una terza; i verbi, tipo "prendere", "nascondere", in una quarta.

In alcuni luoghi, il giocatore può raccogliere oggetti, ad esempio monete d'oro, che conteranno per il punteggio finale. Questi oggetti sono contenuti in una matrice *numerica*, A, seguita da un numero, come A(7); un'altra matrice conterrà il numero di oggetti raccolti. In breve, un gioco d'avventura può essere costituito da una serie di matrici su cui opera il programma.

Lo sviluppo di giochi d'avventura forma oggetto di un'intera serie di lezioni, ma nel frattempo ecco un esempio:

```
10 LET G=14
20 DIM A$(G),A(G)
30 FOR Z=1 TO G
40 READ A$(Z)
50 LET A(Z)=Z
60 NEXT Z
70 FOR X=G TO 2 STEP-1
80 LET Q=RND(X)
90 LET T=A(X):LET A(X)=A(Q):LET A(Q)=
   T
100 NEXT X
110 FOR T=1 TO G:PRINT "LA STANZA□";
    T;"□CONTIENE UN(A)□";A$(A□(T)):
    NEXT T
120 DATA CORDA, SPADA, PINZA,
    COLTELLO, PISTOLA, CHIAVE, TORCIA,
    AUTO, FRUSTA, BACCHETTA, BOMBA,
    LIBRO, MODELLO DI NAVE, ROBOT
```

```
10 LET G=14:PRINT CHR$(147)
20 DIM A$(G),A(G)
30 FOR Z=1 TO G
40 READ A$(Z)
50 LET A(Z)=Z
60 NEXT Z
70 FOR X=G TO 2 STEP-1
80 LET Q=INT(RND(1)*X)+1
90 LET T=A(X):LET A(X)=A(Q):LET A(Q)=
   T
100 NEXT X
110 FOR T=1 TO G:PRINT "LA STANZA";
    T;"CONTIENE UN(A)□";A$(A(T)):NEXT T
120 DATA CORDA, SPADA, PINZA,
COLTELLO, PISTOLA, CHIAVE, TORCIA, AUTO,
FRUSTA, BACCHETTA, BOMBA, LIBRO
130 DATA MODELLO DI NAVE, ROBOT
```

```
10 LET g=14
20 DIM a$(g,10): DIM a(g)
30 FOR z=1 TO g
40 READ a$(z)
50 LET a(z)=z
60 NEXT z
70 FOR x=g TO 2 STEP -1
80 LET q=INT (RND*x)+1
90 LET t=a(x): LET a(x)=a(q): LET a(q)=t
100 NEXT x
110 FOR t=1 TO g: PRINT "La stanza□";
    t;"□contiene un(a)□";a$(a(t)): NEXT t
120 DATA "corda", "spada", "pinza",
    "coltello", "pistola", "chiave", "torcia",
    "auto", "frusta", "bacchetta", "bomba",
    "libro", "modello di nave", "robot"
```

La linea 20 prepara una matrice stringa ed una numerica. La 40 legge una lista di oggetti e la 50 "battezza" gli ambienti. Le linee dalla 70 alla 100 assegnano, a caso, un oggetto a ogni ambiente e la linea 110 visualizza il risultato.

# UN PO' DI PRATICA CON L'ARITMETICA ESADECIMALE

**Sapendo contare su un dito, si può facilmente passare a contare su sedici! Ma, anche senza sedici dita, è semplice esercitarsi con il sistema esadecimale**

Anche se i circuiti dei computer contano in binario, l'uso di un sistema numerico composto esclusivamente di Ø e 1 crea difficoltà a operatori umani, per la gran quantità di zeri che numeri relativamente grandi contengono.

Lunghe serie di Ø e 1 sono difficili da scrivere: è facilissimo commettere errori, che sono praticamente impossibili da individuare in seguito.

Una soluzione, per l'operatore, è l'uso di un sistema numerico con base diversa.

I numeri esadecimali, in forma abbreviata *hex*, sono numeri in base 16, abbastanza simili ai decimali.

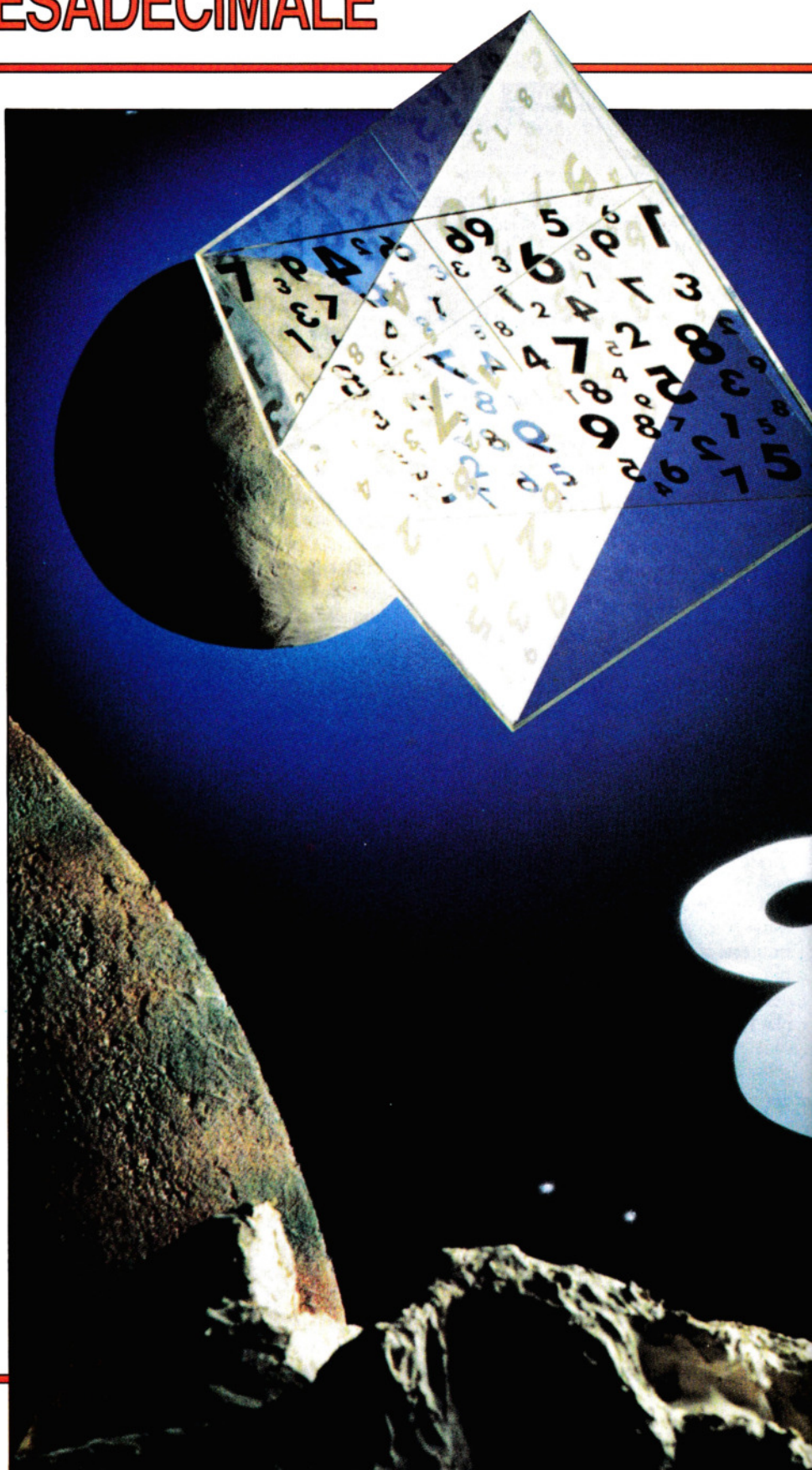
Inoltre, 16 è 2×2×2×2, il che significa che la conversione tra binario e hex è breve e semplice. Il decimale 16 è 1Ø in hex e 1ØØØØ in binario. Ogni numero da Ø a 15 è rappresentato con un numero binario di quattro cifre.

Per usare un sistema numerico con una base maggiore di 1Ø, è necessario definire nuove cifre. In hex, dieci è rappresentato da A, undici da B, dodici da C, tredici da D, quattrodici da E e quindici da F.

## CONVERSIONE DA BINARIO A HEX

Per convertire in hex i numeri binari di otto bit usati dagli home computer, si divide semplicemente il numero in due stringhe di quattro cifre. Poi, come si è visto alle pagine 38 e 39, le prime quattro cifre si riportano direttamente in una cifra hex e le ultime quattro in un'altra cifra hex.

Per esempio, quando si divide 1226 per 16, si ottiene 76 con resto 1Ø; 1Ø è A in hex. 76 si divide per 16 e si ottiene 4 con resto 12. 12 è C in hex e 4 diviso 16 è Ø con resto 4. Perciò 1226 in decimale è 4CA in hex.

Il programma che segue è piuttosto lungo, ma vale la pena di ricopiarlo perché

aiuta a comprendere il funzionamento di questa conversione:

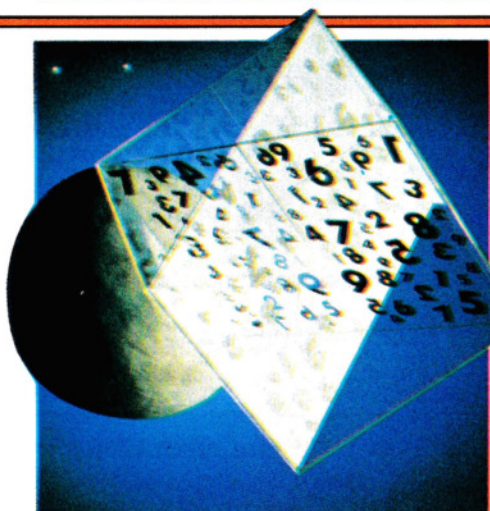
```
20 CLS
25 PLOT 140,0: DRAW 0,160
30 PRINT INVERSE 1;AT 0,8;"□BIN, DEC,
   HEX□"
40 PRINT INVERSE 1;AT 4,2;"□□BINARIO:□
   □□□□"
50 PRINT INVERSE 1;AT 9,2;"□□DECIMALE:
   □□□□"
60 PRINT AT 10,5;"+□□□+□□□+□
   □□+□□□+□□□+□□□+"
70 PRINT INVERSE 1;AT 17,2;
   "□□ESADECIMALE:"
80 PRINT AT 18,4;"+□□+□□+□□="
90 PRINT AT 18,20;"+□□+□□+□□
   ="
100 LET no=0
110 GOTO 150
120 LET a$=INKEY$: IF a$="" THEN
    GOTO 120
130 IF a$="□" THEN LET no=no+1: IF
    no=256 THEN LET no=0
135 IF a$="b" THEN LET no=no-1: IF no
    =-1 THEN LET no=255
140 IF a$="b" OR a$="□" THEN GOTO
    150
145 INPUT "?";no
150 GOSUB 170: GOSUB 250
160 GOTO 120
170 LET nu=no: LET c=128
175 FOR x=0 TO 7
80 LET n=0: IF nu>=c THEN LET n=1:
   LET nu=nu-c
190 LET c=c/2
200 PRINT AT 5,2+4*x;n
210 IF n=1 THEN PRINT AT 10,2+4*x;c*2
220 IF n=0 THEN PRINT AT 10,2
    +4*x;"0□□"
230 NEXT x
235 PRINT AT 13,6;"TOTALE IN DECIMALE
    =□";no;"□□"
240 RETURN
250 LET hi=INT (no/16): LET hh=hi
260 LET lo=(no-hi*16): LET ll=lo: IF lo
    >9 THEN LET lo=lo+7
265 IF hi>9 THEN LET hi=hi+7
270 LET hi=hi+48: LET lo=lo+48
280 PRINT AT 18,14;CHR$ hi;AT 18,30;
    CHR$ lo
290 LET c=8
300 FOR x=0 TO 3
310 LET n=0: IF hh>=c THEN LET n=c:
    LET hh=hh-c
315 LET m=0: IF ll>=c THEN LET m=c:
    LET ll=ll-c
320 LET c=c/2
330 PRINT AT 18,2+x*3;n;AT 18,18+x*3;m
340 NEXT x
400 PRINT AT 21,6;"TOTALE HEX=□";
    CHR$ hi;CHR$ lo
500 RETURN
```

**Conversioni istantanee**

Il BBC B, l'Electron dell'Acorn, il Dragon e il Tandy hanno programmi incorporati per fare conversioni dal decimale all'hex. Per ottenere un numero hex, basta procedere così:

Scrivere PRINT, seguito dal numero decimale desiderato. Poi premere RETURN.

Scrivere PRINT HEX$, seguito dal numero decimale desiderato, racchiuso fra parentesi; ad esempio: PRINT HEX$ (255). Poi premere RETURN.

Se, d'altra parte, volessimo immettere numeri hex come parte di un programma (all'interno di frasi DATA per esempio), questi apparecchi non hanno difficoltà a recepirli. Sugli apparecchi dell'Acorn, il numero hex va preceduto da &, sul Dragon e sul Tandy da &H. I computer convertiranno l'hex in decimale per utilizzarlo nelle operazioni successive.

```
20 CLS
30 PRINT AT 0,9;"BIN, DEC, HEX"
40 PRINT AT 4,4;"BINARIO:"
50 PRINT AT 9,4;"DECIMALE:"
60 PRINT AT 10,5;"+□□□+□□□+□
   □□+□□□+□□□+□□□
70 PRINT AT 17,4;"ESADECIMALE:"
80 PRINT AT 18,4;"+□□+□□+□□="
90 PRINT AT 18,20;"+□□+□□+□□
   ="
100 LET NO=0
110 GOTO 150
120 LET A$=INKEY$
125 IF A$="" THEN GOTO 120
130 IF A$<>"F" THEN GOTO 135
131 LET NO=NO+1
132 IF NO=256 THEN LET NO=0
135 IF A$<>"B" THEN GOTO 140
136 LET NO=NO1
137 IF NO=-1 THEN LET NO=255
140 IF A$="B" OR A$="F" THEN GOTO
   150
145 INPUT NO
150 GOSUB 170
155 GOSUB 250
160 GOTO 120
170 LET NU=NO
171 LET C=128
175 FOR X=0 TO 7
180 LET N=0
185 IF NU>=C THEN LET N=1
186 IF NU>=C THEN LET NU=NU-C
190 LET C=C/2
200 PRINT AT 5,2+4*X;N
210 IF N=1 THEN PRINT AT 10,2+4*X;C*2
220 IF N=0 THEN PRINT AT 10,2+4*X;"0
   □□"
230 NEXT X
235 PRINT AT 13,6;"TOTALE IN DECIMALE
   =□";NO;"□□"
240 RETURN
250 LET HI=INT(NO/6)
255 LET HH=HI
260 LET LO=(NO(HI*16))
261 LET LL=LO
270 LET HI=HI+28
275 LET LO=LO+28
280 PRINT AT 18,14; CHR$ HI; AT 18,30;
   CHR$ LO
290 LET C=8
300 FOR X=0 TO 3
310 LET N=0
311 IF HH>=C THEN LET N=C
312 IF HH>=C THEN LET HH=HH-C
315 LET M=0
316 IF LL>=C THEN LET M=C
317 IF LL>=C THEN LET LL=LL-C
320 LET C=C/2
330 PRINT AT 18,2+X*3;N;AT 18,18+X*3;
   M
340 NEXT X
400 PRINT AT 21,6;"TOTALE IN HEX=□";
   CHR$ HI; CHR$ LO
500 RETURN
```

**D+R**

**Come riconvertire l'hex in decimale?**

Ogni colonna successiva di un numero hex corrisponde a 16 volte la cifra alla sua destra. Perciò per convertire un numero hex come F6DA in decimale, si prende la cifra a destra e la si converte in notazione decimale. A è 10. La cifra accanto a sinistra va convertita in notazione decimale moltiplicandola per 16. D vale 13, per cui: 13×16=208. La cifra successiva va moltiplicata due volte per 16: 6×16×16=1536. Infine, l'ultima cifra va moltiplicata tre volte per 16; poiché F è 15: 15×16×16×16=61440. F6DA in hex è 10+208+1536+6144 o 63194 in decimale. Usando il programma di queste pagine convertire due cifre hex alla volta, moltiplicando poi per 256 le due a sinistra.

```
10 MODE6
20 VDU23;8202;0;0;0;
30 PRINTTAB(13,2)"BIN, DEC, HEX"
40 PRINTTAB(13,3)STRING$(13,CHR$(224))
50 PRINTTAB(5,12);"+□□□+□□□+□
   □□□□+□□□+□□□+□□□+
   □□□="TAB(5,17)"+□□□+□□□
   +□□=□□□□□□□+□□□+□
   □□+□□="
60 PRINTTAB(1,5)"Binario□:"TAB(1,10)
   "Decimale□:"TAB(1,15)"Esadecimale□:"
   TAB(12,20)"NumeroHex□="
70 ?&70=0
80 T=?&70:PROCBIN:PROCDEC:PROCHEX
90 *FX21,0
95 G=GET
100 IF G=32 THEN?&70=?&70+1:GOTO
   80
105 IF G=66 THEN?&70=?&70-1:
   GOTO80
110 PRINTTAB(0,23);:INPUT?&70:PRINTTAB
   (0,23)STRING$(39,"□");:GOTO80
120 DEF PROCBIN
130 FOR X=0 TO 7
140 IF -(T AND 2^X) THEN PRINTTAB(34
   -X*4+(X>3)*2,7)"1"TAB(34-X*4+(X
   >3)*3+(X>6),12);(T AND 2^
   X) ELSE PRINTTAB(34-X*4+(X>3)*
   2,7) "0"TAB(34-X*4+(X>3)*2-
   2,12); "□□0"
150 NEXT X
160 ENDPROC
170 DEF PROCHEX
180 FOR X=4 TO 7
190 PRINT TAB(31-X*4,17);(T AND 2
   ^X)/16
200 NEXT
210 FOR X=0 TO 3
220 PRINTTAB(34-X*4,17);(T AND2^X)
230 NEXT
240 X=(T AND 240)/16
250 A$=CHR$(X+48-7*(X>9))
260 PRINTTAB(18,17);A$
270 X=(T AND 15)
280 B$=CHR$(X+48-7*(X>9))
290 PRINTTAB(37,17);B$
300 PRINTTAB(26,20)A$+B$
310 ENDPROC
320 DEF PROCDEC
330 PRINTTAB(37,12);T"□□"
340 ENDPROC
```

```
20 PRINT "♡"CHR$(8):FOR Z=1 TO
   8:READ A(Z):NEXT Z:DATA
   128,64,32,16,8,4,2,1
30 K$="0123456789ABCDEF":
   POKE 650,255:POKE 53280,0:POKE
   53281,0
40 PRINT "▤"TAB(13)"π BIN,DEC,HEX"
```

```
50 PRINT TAB(13)"■ – – – – – – – – –
   – – –"
60 PRINT "■■■π BINARIO◣:■"
70 PRINT "■■■π DECIMALE◣:■"
80 PRINT "π■■■ESADECIMALE◣:
   ■"
90 PRINT"■◣■■■■■■":V$=
   "": FOR Z=1 TO 8:V$=V$+LEFT$("□"
   +STR$(B(Z))+"□□",4):NEXT Z
100 PRINT MID$(V$,3,14)"□□"RIGHT$(V$,
   16)
110 PRINT "■■■": V$="":FOR Z=
   1 TO 8
120 V$=V$+LEFT$("+"+RIGHT$(STR$(D
   (Z)),LEN(STR$(D(Z)))–1)+"□□□",4):
   NEXT Z
130 PRINT MID$(V$,2,15);"□□□";MID$(V$,
   18,16)"=□□□□■■■■■■";
140 PRINTD(1)+D(2)+D(3)+D(4)+D(5)+
   D(6)+D(7)+D(8)
150 PRINT "■■■":V$="":FOR Z=
   1 TO 8
160 V$=V$+LEFT$("+"+RIGHT$(STR$(H
   (Z)),LEN(STR$(H(Z)))–1)+"□□□",4):
   NEXT Z
170 X1$=MID$(K$,H(1)+H(2)+H(3)+H(4)
   +1,1):X2$=MID$(K$,H(5)+H(6)+H(7)
   +H(8)+1,1)
180 PRINT MID$(V$,2,15);"■=■"X1$"◣
   □□";MID$(V$,18,16);"=□■"X2$
190 PRINT,"■■■■□□TOTALE□HEX
   □:□□□□□□■■■■■■■■■";X1$
   +X2$
200 GET A$:IF A$="" THEN 200
210 POKE 198,0:IF A$="□" THEN A=A
   +1:IF A>255 THEN A=0
220 IF A$="B" THEN A=A–1:IF A<0
   THEN A=255
230 IF A$<>"□" AND A$<>"B" THEN
   350
240 AA=A
250 FOR Z=1 TO 8:IF A(Z)<=AA THEN B
   (Z)=1:D(Z)=A(Z):AA=AA–A(Z):
   GOTO 270
260 B(Z)=0:D(Z)=0
270 NEXT
280 FORZ=1TO4:IFB(Z)=1THENH(Z)=A(Z
   +4):GOTO300
290 H(Z)=0
300 NEXT Z
310 FOR Z=1 TO 4:IF B(Z+4)=1 THEN
   H(Z+4)=A(Z+4):GOTO 330
320 H(Z+4)=0
330 NEXT Z
340 GOTO 90
350 I$="":PRINT "■■■■■■■
   ■■■■■■■■■■■■■
   ■■■■□□□□□□□
   IMMETTI UN NUMERO (0–255):□□□□
   ■■■■■";
360 FOR Z=1 TO 3
370 GET J$:PRINT "*■□■";:IFJ$=
   "" THEN 370
380 IF J$=CHR$(13) THEN 440
390 IF J$=CHR$(20) THEN 350
400 IF ASC(J$)<48 OR ASC(J$)>57
   THEN 370
410 I$=I$+J$:PRINT J$;:NEXT Z
420 GET J$:IF J$=CHR$(20) THEN 350
430 IF J$<>CHR$(13) THEN 420
440 IF VAL(I$)<0 OR VAL(I$)>255
   THEN 310
450 PRINT:PRINT "●□□□□□□□□□□□
   □□□□□□□□□□□□□□□□□□□
   □□□□□□□□□□□□";
460 A=VAL(I$):GOTO 240
```

(C=)

```
10 PRINT"♥"CHR$(8)
20 FOR Z=1 TO 8:READ A(Z):NEXT:DATA
   128,64,32,16,8,4,2,1
30 K$="0123456789ABCDEF":POKE 650,
   255:POKE 36879,8
40 PRINT "■□□□□□π BIN,DEC,HEX"
50 PRINT"□□□□□■ – – – – – – – –
   – – – – –"
60 PRINT "■■π BINARIO◣:"
70 PRINT "■■■π DECIMALE◣:"
80 PRINT "■■■π ESADECIMALE◣:"
90 PRINT "■■■■■■■◣":V$="":
   FOR Z=1 TO 8:V$=V$+STR$(B(Z)):
   NEXTZ
100 PRINT LEFT$ (V$, 8)"□□□" RIGHT$
   (V$, 8)
110 PRINT "■■":V$="":FORZ=1 TO8
120 V$=V$+LEFT$("+"+RIGHT$(STR$(D
   (Z)),LEN(STR$(D(Z)))–1)+"□□□",4):
   NEXTZ
130 PRINT MID$(V$,1,15):PRINTMID$(V$,17(
   16)"■=□□□□■■■■■■■";
140 PRINT D(1)+D(2)+D(3)+D(4)+D(5)
   +D(6)+D(7)+D(8)
150 PRINT "■":V$="":FORZ=1TO8
160 V$=V$+"+"+RIGHT$(STR$(H(Z)),1):
   NEXTZ
170 X1$=MID$(K$,H(1)+H(2)+H(3)+H(4)
   +1,1):X2$=MID$(K$,H(5)+H(6)+H(7)
   +H(8)+1,1)
180 PRINT LEFT$(V$,():"=■"X1$;"□◣"
   RIGHT$(V$,8)"=■"X2$
190 PRINT"■■TOTALE HEX :□□□□□
   □■■■■■■■■■■";X1$+X2$
200 GETA$:IFA$=""THEN200
210 POKE 198,0:IFA$="□"THENA=A+1:
   IFA>255THENA=0
220 IFA$="B"THENA=A–1:IFA<0THENA
   =255
230 IFA$<>"□"ANDA$<>"B"THEN350
240 AA=A
250 FORZ=1TO8:IFA(Z)<=AATHENB(Z)=
   1:(DZ)=A(Z):AA=AA–A(Z):GOTO270
```

```
260 B(Z)=0:D(Z)=0
270 NEXT
280 FORZ=1TO4:IFB(Z)=1THENH(Z)=A(Z
   +4):GOTO300
290 H(Z)=0
300 NEXTZ
310 FORZ=1TO4:IFB(Z+4)=1THENH(Z+
   4)=A(Z+4):GOTO330
320 H(Z+4)=0
330 NEXTZ
340 GOTO90
350 I$="":PRINT"
IMMETTIUNNUMERO (0-255) >
   ";
360 FORZ=1TO3
370 GETJ$:PRINT"  ";:IF J$=""
   THEN370
380 IF J$=CHR$(13)THEN440
390 IFJ$=CHR$(20)THEN350
400 IFASCJ$)<48 ORASC(J$)>57THEN370
410 I$=I$+J$:PRINTJ$;:NEXTZ
420 GETJ$:IFJ$=CHR$(29) THEN350
430 IFJ$<>CHR$(13) THEN420
440 IFVAL(I$)<0ORVAH(I$)>255THEN350
450 PRINT:"
   ";
460 A=VALI$):GOTO240
```

**Ecco come vengono visualizzati i numeri Bin, Dec, Hex sullo Spectrum. L'ordine sullo schermo non è molto diverso sugli altri apparecchi. Un confronto tra i tre diversi sistemi numerici è immediato. Dopo il RUN, le tre linee sono poste a zero. Premendo B la linea del Binario si riempie di 1. Al rigo sotto, la linea del decimale si riempie delle potenze di 2. Da destra a sinistra, si ha: 1, cioè $2^0$; 2, cioè $2^1$; 4, cioè $2^2$ ecc. La linea hex funziona analogamente, ma le due cifre hex sono calcolate indipendentemente.**

```
20 CLS0
30 PRINT@11,"BIN, DEC, ESA";
40 PRINT@68,"BINARIO";
50 PRINT@196,"DECIMALE";
60 PRINT@323,"ESADECIMALE";
70 PRINT@355,"+□□+□□+□□□=
   □";
80 PRINT@371,"+□□+□□+□□□=
   □";
90 FORJ=1TO15:POKE1040+32*J,175:
   NEXT
100 PRINT@450,"NUMERO ESA =
   □□□□□";
110 PRINT@227,"+□□□+□□□+□□
   □□+□□□+□□□+□□□+"
120 GOTO170
130 IN$=INKEY$:IFIN$="" THEN130
140 IFIN$="□"THENNO=NO+1:NO=
   NO AND 255:GOTO170
150 IFIN$="B" THENNO=NO-1:NO=
   NO AND 255:GOTO170
160 GOSUB 370
170 GOSUB190:GOSUB270
180 GOTO130
190 FORX=7TO0 STEP-1
200 IF (NO AND 2 ↑ X) THENN = 1 ELSE
   N=0
210 PRINT@125-X*4,N;
220 IFN=1 THENN=INT(2 ↑ X):N$=STR$
   (N):N$=MID$(N$,2,LEN(N$)-1)ELSEN$
   =RIGHT$("□□0', LEN(STR$(2 ↑ X))-1)
230 PRINT@255-X*4-LEN(N$),N$;
240 NEXT
250 PRINT@279,"□=□";MID$(STR$(NO)+
   "□□",2,3);
260 RETURN
270 FORX = 7 TO 4 STEP-1
280 PRINT@374-X*3,STR$((NO AND 2 ↑
   X)/16);
290 NEXT
300 PRINT@367,HEX$(NO/16);
310 FORX=3TO0 STEP-1
320 PRINT@378-X*3,STR$(NO AND 2 ↑ X);
330 NEXT
340 PRINT@383,HEX$(NO AND 15);
350 POKE1488,PEEK(1391):POKE1489,PEEK
   (1407)
360 RETURN
370 NU$="":PRINT@439,"?";
380 IN$=INKEY$:IF(IN$<"0"ORIN$>
   "9") ANDIN$<>CHR$(13)
   THEN GOTO 380
390 IFIN$=CHR$(13) THENNO=VAL(NU$):
   IFNO>255 THEN370ELSEPRINT@439,
   STRING$(5,CHR$(128));:RETURN
400 IFIN$<>CHR$(13) ANDLEN (NU$)>
   2 THEN380
410NU$=NU$+IN$:PRINT@441,MID$(NU$
   +"□□□",1,3);:GOTO380
```

Trascritto il programma per il proprio apparecchio e eseguito un RUN, si vede che i numeri binari, decimali e hex sono tutti posti a zero. Premendo la barra spaziatrice, il valore viene aumentato di 1 in ogni base; si continui a premere e il computer andrà avanti a contare, aggiungendo un 1 al totale.

Si noti come l'equivalente decimale è calcolato aggiungendo il valore di ogni colonna che contiene un 1 nel binario.

L'hex si calcola in modo analogo, salvo che si prendono quattro cifre binarie per volta.

Premendo il tasto B si sottrae un 1 da ogni numero e si va all'indietro.

Per lo Spectrum e i Commodore, può essere utile conservare il programma, poiché consente anche la conversione di numeri decimali in hex. Il modo più rapido per una conversione è premere qualsiasi tasto sulla tastiera, meno la barra spaziatrice o B. Compare sullo schermo un punto interrogativo: immettendo un qualsiasi numero decimale minore di 225, e premendo ENTER o RETURN, viene visualizzato l'equivalente binario e hex.

Si noterà che il numero più grande rappresentabile in binario, con otto bit, è 11111111. Esso corrisponde a 255 in decimale e ad FF (il numero massimo di due cifre) in hex. Ogni numero memorizzato in un byte di un microcomputer si può rappresentare con un numero hex di due cifre e il linguaggio macchina è interamente costituito da simili codici hex di due cifre.

## NUMERI PIU' GRANDI

Il computer elabora numeri più grandi di 255 semplicemente dividendoli in due parti e depositandoli in due locazioni di memoria adiacenti. Ciò consente di memorizzare ogni numero fino a FFFF in hex o 65.535 in decimale. FFFF è un numero importante negli home computer, dato che è il numero massimo per gli indirizzi delle locazioni di memoria. Possono venir memorizzati numeri ancor più grandi, se si scompone ogni numero in tre o quattro byte esadecimali e li si deposita in locazione di memoria successive. Il modo in cui i byte vengono memorizzati è un problema di convenzioni: i computer Sinclair, Commodore e Acorn memorizzano il byte con il valore più basso nella locazione di memoria più bassa e il più alto in quella più alta, mentre il Dragon e il Tandy nel modo opposto.

In che modo rappresentare in hex i numeri negativi, sarà argomento della prossima lezione.

# INDICE CUMULATIVO DEI FASCICOLI

# NEL PROSSIMO NUMERO

☐ *Affiniamo le nostre capacità nel disegno, imparando nuovi COMANDI GRAFICI del BASIC ed esercitandoci nei loro impieghi più efficaci.*

☐ *Zap...pum...crash. Ricreare graficamente l'effetto di ESPLOSIONI è particolarmente utile in un'infinità di giochi. Ecco come ottenere simili effetti visivi.*

☐ *Scopriamo un nuovo affascinante aspetto dei numeri usati nel codice macchina: come il computer interpreta i NUMERI NEGATIVI.*

☐ *Cosa s'intende per programmi "ben" scritti o "eleganti"? La risposta la troviamo nella lezione sulla PROGRAMMAZIONE STRUTTURATA.*

☐ *Inoltre, per i possessori di Commodore, la prima di una serie di lezioni sulla GRAFICA A SPRITE.*